UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITECNICA SUPERIOR

DESARROLLO DE UN ENTORNO DE USUARIO PARA APLICACIÓN DE REDES BAYESIANAS DINÁMICAS A PROBLEMAS DE FUSIÓN DE INFORMACIÓN

PROYECTO FIN DE CARRERA
INGENIERÍA INFORMÁTICA

Autor: Daniel García Gallego
Tutor: Jesús García Herrero

FEBRERO 2010

Resumen:

El presente proyecto de fin de carrera trata los conceptos relacionados con la construcción de un sistema experto con incertidumbre a través de la representación gráfica de un modelo probabilísticos temporal, y más concretamente, a través de la técnica de las redes bayesianas dinámicas. El objetivo es presentar al lector la información necesaria que pueda ser de ayuda para futuras aplicaciones a través del uso de redes bayesianas, recogiendo todos los aspectos teóricos, mostrando los distintos tipos de algoritmos y los conceptos relacionados que se pueden emplear en distintas situaciones o casos. También, se desarrollará un entorno de usuario que nos permitirá trabajar con modelos probabilísticos en el tiempo a través de la ejecución de una red bayesiana dinámica, que nos realizará el tratamiento del razonamiento temporal con incertidumbre, aplicado en la formalización de distintas aplicaciones de comprensión de situaciones. Por último, se expondrán unos casos experimentales que nos permitan evaluar la aplicación, donde se detallarán los resultados obtenidos, los cuales, nos permitirán presentar los aspectos teóricos tratados.

Abstract:

This dissertation exposes the concepts associated with the construction of an expert system with uncertainty through the graphical representation of a temporal probabilistic model and, specifically, through the technique of dynamic bayesian belief networks. The aim is to provide the reader with the necessary information that may be helpful for future applications through the use of bayesian networks, gathering all theoretical features, showing the different types of algorithms and concepts that can be used in different situations or cases.Moreover, we will develop a user environment which will allow us to work with probabilistic models in time through the execution of a bayesian belief network, that will process the temporal treatment of the reasoning under uncertainty, applied to the formalization of different applications for the comprehension of situations.Finally, we will expose some practical cases which will allow us to evaluate the application developed, where the results will be specified, helping to illustrate the theorical aspects.

Agradecimientos

A mis padres por su apoyo y paciencia lo largo de la carrera.
A los profesores que a lo largo de la carrera han ejercido de fuente de conocimiento y en especial a Jesús García por su orientación y ayuda a lo largo de este proyecto.
Y en resumen, a todas las personas que directamente o indirectamente, han contribuido a la realización de este trabajo.

Índice de Contenidos

Índice de figuras

Índice de imágenes

Índice de tablas

Capítulo I. Introducción

Antes de la existencia y desarrollo de las redes bayesianas se utilizaban sistemas expertos probabilistas, los cuales requerían mucha información y demasiados cálculos complejos para la resolución de problemas en los que interviniesen un elevado número de variables, por lo que, en la mayoría de las ocasiones, eran desestimados. No obstante, todo esto cambió a partir de los años 80 con una publicación "*Probabilistic Reasoning in Intelligent Systems"* del grupo del profesor Judea Pearl, en la Universidad de California-Los Ángeles (UCLA), en el cuál desarrolla un modelo gráfico para la construcción de sistemas expertos probabilistas en inteligencia artificial (1).

Las ideas sobre las que se basaba en su trabajo, consistían en aprovechar las relaciones de dependencia e independencia, existentes entre las variables de un problema, antes de calcular y especificar los valores numéricos de las probabilidades, reduciendo considerablemente el número de cálculos. Siendo posteriormente representadas dichas relaciones a través de modelos gráficos.

A partir de la publicación de Pearl, surgieron una serie libros entre otros Neapolitan (2), Lauritzen (3), Jensen (4) y Jordan (5) que continuarían con el asentamiento de las bases del modelo probabilístico de las redes bayesianas.

De modo paralelo a la aparición de las publicaciones surgen en los años 90 una serie de grupos de investigación en las universidades más prestigiosas (Stanford, Harvard, UCLA, Massachusetts Institute of Technology,…), así como en la mayoría de empresas líderes en el mundo de la informática. Actualmente, existen una serie de investigaciones sobre modelos probabilísticos temporales basados en la utilización de redes bayesianas dinámicas, que son el paradigma más reciente de las redes bayesianas y general, aplicable a muchos problemas que habían estado limitado por los requisitos computaciones, pero que poco a poco, se están resolviendo (6).

En España, hay principalmente tres únicos grupos que están especializados en este campo. El primero de ellos está dirigido por Serafín Moral y Luis de Campos en la Universidad de Granada. El segundo, está compuesto por Enrique Castillo y José María Sarabia en la Universidad de Cantabria. Y el último por los profesores Pedro Larrañaga y Jose A. Lozano de la Universidad del País Vasco. Siendo casi todos, autores matemáticos especializados en probabilidad y estadística. Aparte de estos grupos, existen varios investigadores en Madrid y Barcelona que han estudiado ocasionalmente, las redes bayesianas y su relación con otros modelos.
En 1996 varios grupos de universidades españolas se unieron para solicitar un proyecto de investigación, que se desarrollo entre 1997 y 2000. Su principal logro, a parte de las publicaciones a las que dieron lugar sus investigaciones, consistió en el desarrollo del programa Elvira, una herramienta implementada en el lenguaje Java y destinada a la edición y evaluación de modelos gráficos probabilistas, concretamente redes bayesianas y diagramas de influencia.
Debido al éxito obtenido en el proyecto, llevó a los investigadores a solicitar un nuevo proyecto en marzo 2001, titulado "*Elvira II: Aplicaciones de los Modelos Gráficos*

Probabilísticos", que fue concedido por el Ministerio de Ciencia y Tecnología. Siendo sus objetivos principales dos: mejorar las características del programa Elvira actual y desarrollar aplicaciones en diversos campos, como la medicina, la genética, la agricultura y el comercio inteligente (7).
La aplicación de las redes bayesianas en el entorno tecnológico, como útiles informáticos, aparecen en la década de los años noventa, dando lugar a numerosas publicaciones sobre sus fundamentos y aplicaciones.
En cuanto a las aplicaciones, aparecen explícitamente en gestión y en toma de decisiones en general y de forma específica en análisis de riesgos, análisis de fiabilidad y también, en el desarrollo de sistemas expertos. Como referencias de empleo de las redes bayesianas en distintos apartados como son el reconocimiento de situaciones, se pueden citar *"Situation Assessment via Bayesian Belief Networks"* (8), *"Video Understanding for Metro Surveillance"* (9)*, "Bayesian Methods for Image Super-Resolution"* (10)*, "Exploiting Human Actions and Object Context for Recognition Tasks"* (11)*, Automatic recognition of facial expressions using Bayesian Belief Networks* (12).

Además de las publicaciones citadas anteriormente, un reducido número pero significativo, han surgido a lo largo de estos años una gran cantidad de productos software[1], tanto comercial como académico, debido a la creciente utilización de las redes bayesianas en los distintos aspectos de la inteligencia artificial, tanto en los sectores educativos como en las empresas.

Como ejemplos típicos de las redes bayesianas aplicada en sistemas expertos, se puede citar:

- El sistema CPCS-BN para medicina interna (13), que consta de 448 nodos y 908 arcos, y que ha sido evaluado favorablemente por los principales médicos del mundo en medicina interna.
- En el proyecto *Lumiere*, de Microsoft® Research, ha desarrollo una red bayesiana basada en el objetivo que los usuarios querían obtener a partir de sus acciones (14).
- ProstaNet es una red bayesiana para el diagnóstico de cáncer de próstata (15). Su construcción manual se llevó a cabo en conjunto con un urólogo, con un avance de 5 versiones hasta el momento. Consta de 47 nodos: 7 enfermedades o anomalías. 6 síntomas, 8 signos, 10 pruebas, 11 factores de riesgo, 1 tratamiento y 4 auxiliares.
- Diagnóstico de fallas de Procesador, Intel® está utilizando redes bayesianas para diagnóstico de chips desde 1990 (16). Utilizan un conjunto de pruebas que infieren posibles problemas del proceso. Después de algunos años de evolución, el sistema ya es estable. Utiliza unos pocos cientos de nodos en una jerarquía de tres niveles, donde la mayor dificultad fue la obtención de las probabilidades *a priori*.

La muestra de todo este conjunto de sistemas expertos y del conjunto de herramientas disponibles es, solamente, una muestra de la gran expansión que están teniendo las

[1] En el Anexo I: Software disponible se puede consultar el resumen de las herramientas que implementan las técnicas y funcionalidades de las redes bayesianas.

redes bayesianas en el mundo. Así, uno puede observar la importancia y el rápido asentamiento, si se considera como fecha de inicio la aparición de la publicación de Pearl, de esta nueva rama de la inteligencia artificial.

1.1. Contexto

El marco donde se encuentra ubicado este proyecto es en el Departamento de Informática de la Universidad Carlos III, y más concretamente en el Grupo de Inteligencia Artificial Aplicada (GIAA).

Uno de los motivos por el cual se ha desarrollado este proyecto tiene su origen en los problemas con incertidumbre y en la manera de representarla. Permitiendo así, desarrollar un proceso práctico, que sirva de modelo a posibles estudios posteriores, sobre la técnica de redes bayesianas en cualquier ámbito.

1.2. Finalidad del proyecto

El principal objetivo para el desarrollo de este proyecto ha sido el poder abordar el problema de análisis de situaciones reales en el modelado de sistemas expertos de clasificación o predicción a través de la utilización de redes bayesianas dinámicas y pasando por los distintos niveles de fusión. Además, se exhibirá en tipo real la representación gráfica de los hallazgos producidos por las distintas variables de observación, dando lugar a la propagación de la certidumbre y la obtención del resultado esperado. Con el desarrollo del entorno de usuario se podrán realizar, a través de un proceso continuo, razonamientos en el tiempo con flujos de información sensorial (o de otro tipo) que llegan de una manera secuencial al sistema. Siendo esta una de la diferencias respecto de los problemas clásicos de clasificación y de sistemas expertos.

Las redes serán construidas teniendo en cuenta los datos provenientes de la clasificación del problema y en la medida de lo posible, de expertos en el dominio del contexto.

El sistema final será presentado con un doble objetivo. Por un lado, a nivel teórico, para entender el proceso de construcción y elaboración de redes bayesianas para cualquier tipo de problema con incertidumbre que se pueda tratar como modelos probabilísticos. Y de otro lado, a nivel práctico en la representación visual, estudio y exploración de casos prácticos, que nos permita evaluar los usos de las redes bayesianas dinámicas en el análisis de situaciones temporales.

1.3. Estructura de la memoria

El documento se encuentra dividido en siete capítulos más tres anexos, que se detallarán a continuación:

- **Capítulo 1. Introducción.** Se plantearán las motivaciones de la realización del proyecto, explicando su contexto y la finalidad del mismo. Se realiza una pequeña visión a las historia de las redes bayesianas así como una pequeña

muestra de la utilización de esta técnica en sistemas expertos actuales. También se realizarán las definiciones previas necesarias para la lectura y compresión del resto del trabajo. Por último se expondrá la estructura del documento que se detalla en esta sección y se presentarán las herramientas empleadas en el desarrollo del proyecto.

- **Capítulo 2. Estado del arte.** Se presentará una introducción a los sistemas expertos. Se comenzará con algunas definiciones de sistemas expertos con algún ejemplo que muestre la importancia y la amplia aplicabilidad de los mismos. Se discutirá y se analizará sobre la estructura de estos sistemas y sus principales componentes. Por último se mostrarán las diferentes etapas necesarias para el diseño, desarrollo e implementación de los sistemas expertos. También se expondrá en este capítulo el conjunto de programas necesarios que han sido utilizados para el desarrollo del proyecto.

- **Capítulo 3. Redes bayesianas.** Se introducirán todos los conceptos necesarios para el entendimiento de las redes bayesianas. Se definirá la sintaxis y semántica de estas redes y se mostrará cómo se utilizan para capturar conocimiento incierto de un modo natural y eficiente. Se presentarán los distintos tipos de redes bayesianas con sus cualidades y beneficios, se describirán los diversos algoritmo de inferencia así como técnicas de aprendizaje y clasificadores basados en las distintas redes bayesianas según su grafo. El tema principal de este capítulo son los conceptos de las redes bayesianas dinámicas sobre las que se basa el desarrollo del entorno de usuario que nos permitirá el tratamiento de problemas de fusión de información. Es de señalar, que aunque no se desarrollarán ningún capítulo en este proyecto sobre la sintaxis y semántica de la teoría de la probabilidad, sí que es recomendable tener un cierto conocimiento del mismo para un mejor entendimiento.

- **Capítulo 4. Objetivos.** Este apartado representa el análisis de objetivos de la interfaz a desarrollar, recogiendo la especificación de los casos de uso y exponiendo los requisitos de usuario.

- **Capítulo 5 Desarrollo de la solución.** Abarcará el desarrollo completo de la aplicación y se describirán los pasos que se han ido realizando en cada fase de su implementación. Los métodos realizados en la construcción del sistema se basarán sobre los aspectos teóricos presentados en el capítulo 3. Redes Bayesianas y se emplearán para ello las herramientas dispuestas en el presente documento. Por último, en este capítulo también se presentarán las decisiones tomadas en los puntos claves del desarrollo, así como la explicación de la medida adoptada.

- **Capítulo 6. Experimentación.** Se presentarán los resultados obtenidos a través de casos prácticos que nos permitan evaluar la interfaz desarrollada exponiendo los aspectos más importantes encontrados y haciendo hincapié en las conclusiones más destacas de cada análisis realizado.

- **Capítulo 7. Conclusiones.** Este apartado presenta el estudio final del desarrollo del proyecto, donde se exponen las conclusiones y objetivos obtenidos. También se realiza el análisis de los casos prácticos así como el estudio de líneas de futuros trabajos.

- **Capítulo 8. Gestión del proyecto.** En este capítulo final de proyecto, se recogen las cuestiones relacionadas con la gestión del mismo desde el punto de vista de la Ingeniería del Software. En él, se recogen cuestiones como la planificación realizada, los recursos empleados, la tecnología usada y una estimación del coste del mismo.

- **Anexo A. Software disponible.** En este anexo se mostrará un resumen de la amplia disponibilidad de software que implementa la técnica de redes bayesianas, recogiendo sus características y el lugar disponible para su localización. Es pues una muestra para posibles trabajos futuros que necesiten de un software específico que concuerde con sus necesidades de requisitos.

- **Anexo B. Manual de instalación.** Contendrá el proceso de instalación del sistema desarrollado así como de las herramientas y/o programas utilizados durante el proyecto. Se contemplarán y se especificarán las configuraciones de los programas así como del entorno de desarrollo.

- **Anexo C. Manual de usuario.** En el presente anexo, se recoge el manual de usuario del programa Netica, así como de la interfaz desarrollada que sirva de documento de consulta para futuros usuarios que quieran consultar las funciones o procesos más utilizados en el proyecto.

1.4. Acrónimos y definiciones

1.4.1. Acrónimos

- **API**: Interfaz de Programación de Aplicaciones, del inglés *Application Programming Interface*. Es un conjunto de rutinas que provee un sistema operativo, una aplicación o una biblioteca, que definen cómo invocar desde un programa un servicio que éstos prestan. En resumen una API representa una interfaz de comunicación entre componentes software.

- **COM**: Componet Object Model es una plataforma de Microsoft para compoenentes de software. Esta plataforma es utilizada para permitir la comunicación entre procesos y la creación dinámica de objetos, en cualquier lenguaje de programación que soporte dicha tecnología.

- **GUI**: Interfaz Gráfica de Usuario, del inglés *Graphic User Interface*. Conjunto de formas y métodos que posibilitan la interacción de un sistema con los usuarios utilizando formas gráficas e imágenes. Con formas gráficas se refiere a botones,

iconos, ventanas, fuentes, etc. los cuales representas funciones, acciones e información.

- **IA**: Inteligencia Artificial.

- **UML**: Lenguaje Unificado de Modelado, del inglés *Unified Modeling Language*, es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

1.4.2. Definiciones

- **Antepasado o ascendiente:** X es un antepasado o ascendiente de Z si y sólo si existe un camino dirigido de X a Z.

- **Árbol:** es un poliárbol en el que cada nodo tiene un solo padre, menos el nodo raíz que no tiene.

- **Arco:** es un par ordenado de nodos {X, Y}. En la representación gráfica un arco vienen dado por una flecha desde X hasta Y (X→Y).

- **Ciclo:** es un camino no dirigido que empieza y termina en el mimo nodo X.

- **Clique:** clique es un subconjunto de nodos completamente conectados máximo, de forma que hay un arco entre cada par de nodos, y no existe un conjunto completamente conectado del que éste sea subconjunto.

- **Descendiente:** Z es un descendiente de X si y sólo si X es un antepasado de Z.

- **Diagrama Gantt:** es una herramienta gráfica cuyo objetivo es el de mostrar el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado. A pesar de que, en principio, el diagrama de Gantt no indica las relaciones existentes entre actividades, la posición de cada tarea a lo largo del tiempo hace que se puedan identificar dichas relaciones e interdependencias.

- **Evidencia:** es un conjunto de hallazgos de a = $\{A = a_1,\ldots, A_i = a_i\}$.

- **Experto:** es la persona que interactúa con el ingeniero del conocimiento, aportando sus conocimientos y experiencia de un área particular del saber humano.

- **Framework:** es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje de scripting entre otros tipos de software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

- **Grafo acíclico dirigido:** conocido como DAG por sus siglas en inglés (Directed Acyclic Graph), es un grafo dirigido que no tiene ciclos. Para cada vértice *V*, no hay un camino directo que empiece y termine en *V*.

- **Grafo conexo:** cuando entre cualquier par de nodos hay al menos un camino no dirigido.

- **Grafo múltiplemente conexo:** contiene bucles o ciclos.

- **Hallazgo:** es la determinación del valor de una variable $A = a_1$ a partir de un datos de una observación.

- **Independencia**: dos variables X e Y son independientes si se tiene que P (X|Y) = P(X). De esta definición se tiene una caracterización de la independencia que se puede utilizar como definición alternativa: X e Y son independientes si y solo si P(X | Y) = P(X) * P(Y).

- **Independencia condicional**: dos variables X e Y son independientes dado una tercera variable Z si se tiene que P(X|Y;Z) = P(X|Y). De esta definición se tiene una caracterización de la independencia que se puede utilizar como definición alternativa: X e Y son independientes dado Z si y solo si P(X; Y|Z) = P(X|Z) * P(Y|Z). También se dice que Z separa condicionalmente a X e Y .

- **Ingeniero del conocimiento:** es el especialista, que entrevista al experto cuya función consiste en extraer el conocimiento de los expertos humanos en un determinado área, y en codificar dicho conocimiento de manera que pueda ser procesado por un sistema.

- **Padre:** X es padre de Y si y sólo si existe un arco X→Y. También se dice que Y es hijo de X.

- **Poliárbol o grafo simplemente conexo:** cuando entre cualquier par de nodos hay un único camino. Son gráficos sencillamente conectados en el que un nodo puede tener más de un padre.

- **Probabilidad *a priori*:** es la probabilidad de una variable o subconjunto de variables cuando no hay ningún hallazgo. Coincide con la probabilidad marginal *P(X)*.

- **Probabilidad a posteriori:** es la probabilidad de una variable o subconjunto de variables dada la evidencia *e*. Se trata de la probabilidad condicional *P(x|e)*.

- **Nodo terminal**: es un nodo que no tiene hijos.

- **Separación:** El conjunto Z separa los conjunto X e Y si estos dos últimos son condicionalmente independientes dado Z.

1.5. Entorno de trabajo

El entorno de trabajo utilizado en el proyecto se compone de las aplicaciones empleadas durante el desarrollo y por las herramientas utilizadas para la generación de la documentación.
Las aplicaciones y herramientas utilizadas han sido las siguientes:

1.5.1. Microsoft Visual Studio 2005

La construcción del proyecto se ha realizado bajo este entorno de desarrollo, empleando como lenguaje de programación Visual C++.

1.5.2. Aplicación Netica y API

Para la ejecución de la parte práctica desarrollada se ha hecho uso de este programa que cuenta con la interfaz gráfica para la visualización de los casos de prueba, además han sido necesarias las librerías correspondientes para la implementación, creación, modelado y ejecución de las redes bayesianas.

1.5.3. Microsoft Project

Este software de administración de proyectos ha sido empleado para realizar la planificación, la asignación de tiempos y los recursos del mismo.

1.5.4. Microsoft Office

Para la realización de la memoria del proyecto, de la obtención de los gráficos, presupuesto y presentación, se han empleado algunas de las utilidades incluidas dentro de la suite ofimática Microsoft Office 2003 tales como Microsoft Word, Microsoft Power Point, Microsoft Excel y Microsoft Visio.

1.5.5. Microsoft Windows XP

El desarrollo de todo el proyecto junto con la documentación aportada, ha sido llevado a cabo bajo este sistema operativo.

1.5.6. Gnuplot

Programa flexible de generación de funciones y datos que se emplea para la visualización de los gráficos generados por la interfaz.

Capítulo II. Revisión de los sistemas expertos

A lo largo de este proyecto, se utilizarán diversos enfoques de resolución del problema que cubren temas muy diversos como transformaciones de dominio, aplicación de distribuciones estadísticas, uso de técnicas de clasificación automática o filtrado de datos, siendo cada uno de estos temas enormemente amplios en sí mismo. Es en este capítulo, donde se intentará recoger los aspectos teóricos y la metodología a emplear que se basará en las fases que componen la estructura básica de los sistemas expertos.

2.1. Sistemas expertos

El tratamiento de la incertidumbre constituye uno de los campos fundamentales de la inteligencia artificial, pues afecta en mayor o menor medida a todos los demás. En particular, una de las propiedades esenciales de los sistemas expertos, y a la vez una de las más complejas, es el tratamiento de la incertidumbre. Es pues, esta incertidumbre, la que hace que los sistemas expertos, constituyentes de una de las áreas de investigación en el campo de la IA, estén presentes en la mayor parte de las restantes áreas, ya que en mayor o menor grado disponen de este sistema como un componente formando parte de ellos.

2.1.1. ¿Qué son?

El concepto de sistema experto fue formulado por primera vez en 1982 por Edward A.Feigenbaum en el "*International Joint Conference on A.I.*" y que definió posteriormente un sistema experto como:

> «… un programa inteligente para ordenador que utiliza conocimiento y procedimiento inferenciales en la resolución de problemas, problemas que son lo suficientemente difíciles como para que su solución requiera una experiencia humana importante. El conocimiento necesario para actuar así, junto con los procedimientos inferenciales utilizados, puede considerarse como un modelo de la experiencia de los mejores expertos campo.
> El conocimiento de un sistema experto está compuesto por hechos y por heurísticos. Los «hechos» constituyen un cuerpo de información ampliamente compartido, públicamente disponible, y sobre el cual, generalmente, los expertos del campos están de acuerdo. Los «heurísticos» son básicamente personales, son reglas de buen juicio no muy bien estudiadas (reglas de razonamiento plausible, reglas de buenas conjeturas) que caracterizan la toma de decisiones a nivel de experto en el campo. La calidad de las prestaciones de un sistema experto es, fundamentalmente, función del tamaño y de la calidad de las base de conocimientos que posee (17)».

Aunque la anterior definición marcó el comienzo del desarrollo de esta área dentro de la inteligencia artificial, es de señalar que han surgido desde entonces otras definiciones. Como por ejemplo:

- "es un sistema de cómputo que **emula** la habilidad de tomar decisiones de un especialista humano. El término *emular*[2] significa que el sistema experto tiene el objetivo de actuar en todos los aspectos como un especialista humano (18).

- "son programas de ordenador diseñados para actuar como un especialista humano en un dominio particular o área de conocimiento. En este sentido, pueden considerarse como intermediarios entre el experto humano, que transmite su conocimiento al sistema, y el usuario que lo utiliza para resolver un problema con la eficacia del especialista. El sistema experto utilizará para ello el conocimiento que tenga almacenado y algunos métodos de inferencia (19)."

- "un programa de ordenador que resuelve problemas que requieren experiencia humana, mediante el uso de representación del conocimiento y procedimientos de decisión" (20)".

En este documento se considerará como sistema experto, *como un sistema informático (hardware y software) que simula a los expertos humanos en un área de especialización dada* (21).

Por consiguiente es de esperar que un sistema experto mejore la productividad de un experto humano en un campo particular. Que ahorre tiempo y dinero, permitiendo conservar su valioso conocimiento y difundirlo más fácilmente.

En la figura 2.1, se puede observar los campos dominantes que emplean sistemas expertos. Estos resultados están basados en la investigación de unos 2500 sistemas expertos que realiza Durkin (22).

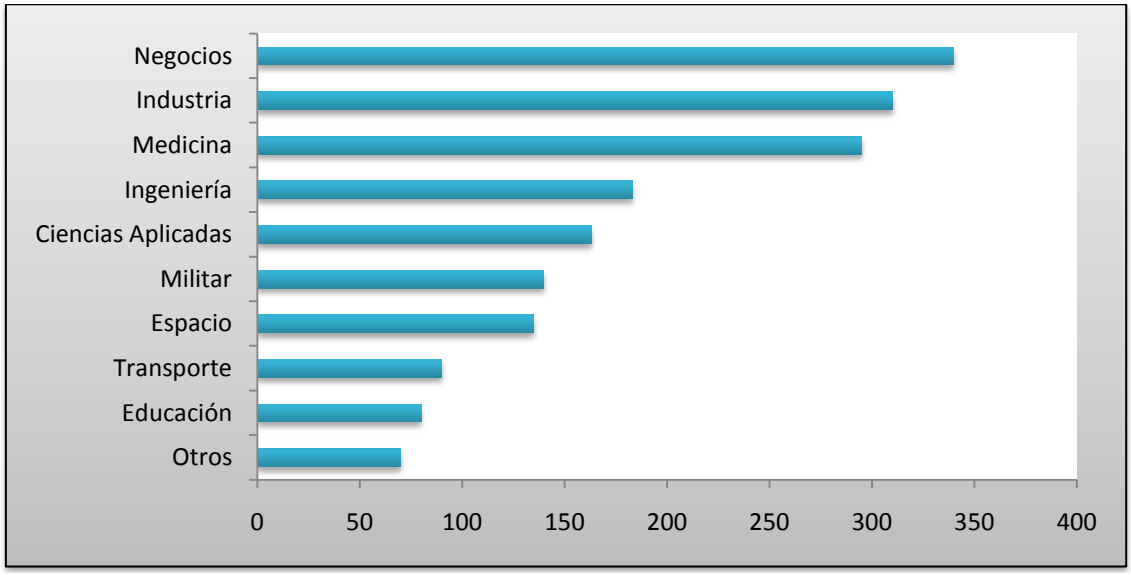

Figura 1.1. Campos de aplicación de los sistemas expertos.

[2] Una emulación es mucho más fuerte que una simulación, que en algunos aspectos sólo requiere que se actúe como en la realidad.

2.1.2. ¿El por qué de estos sistemas?

Como hemos comprobado anteriormente, existe una gran cantidad de áreas de aplicación distintas que hacen uso de los sistemas expertos. Esto nos hace plantearnos el por qué del uso de estos sistemas y no de otros. Aunque existen varias razones para la utilización de los sistemas expertos, las más importantes son:

1. Pueden usarse en ambientes poco hostiles que, en caso de no implantarse, podrían ser peligrosos para un ser humano dañando su integridad física.

2. Aunque el desarrollo o la adquisición de un sistema experto es generalmente caro, se consigue un gran beneficio por el bajo coste que se obtiene del mantenimiento y de su uso repetido.

3. Ganancias muy altas en términos monetarios, precisión y tiempo como resultado del uso, siendo la amortización muy rápida.

4. Hay situaciones que se pueden dar en las que sea necesaria una respuesta rápida, o en tiempo real. Dependiendo del software y hardware usado, un sistema experto puede responder más rápido y estar más dispuesto que un experto humano, haciendo por consiguiente, que el sistema sea muy valiosos en los casos en los que el tiempo de respuesta sea crítico.

5. El conocimiento del sistema experto es permanente. La gran diferencia con los expertos humanos, es que estos pueden retirarse, renunciar o morir perdiendo por consiguiente, el conocimiento adquirido a lo largo de sus años de experiencia.

6. Varios sistemas expertos pueden combinarse con otros para dar lugar a sistemas más fiables, ya que se combina la sabiduría colectiva de varios expertos humanos en lugar de la de uno solo. El nivel de experiencia combinada de muchos sistemas expertos puede exceder el de un solo especialista humano (23).

7. Las respuestas proporcionadas por el sistema experto son sólidas, completas y sin emociones en todo momento. Esta característica hace que el sistema sea importante en tiempo real y en situaciones de emergencia, cuando un experto posiblemente no funcionaría en toda su capacidad a causa de la fatiga o de la presión.

8. Los sistemas expertos además de poder realizar operaciones monótonas e inconfortables para los humanos, son capaces de realizar respuestas rápidas y aproximadas cuando la complejidad de muchos problemas impiden al experto humano resolverlo. Es en estos casos donde la capacidad de procesamiento y el elevadísimo número de operaciones complejas realizadas por los sistemas expertos proporcionan la capacidad de inducir respuestas fiables.

9. A través del uso de los sistemas expertos, el personal con poca experiencia puede resolver problemas que requieran un conocimiento experto. Además es de señalar que el uso de dichos sistemas puede ayudar a la capacitación y adiestramiento del personal sin experiencia, donde el número de personas con acceso al conocimiento aumenta con el uso de los sistemas expertos.

Aunque se han visto una serie de beneficios, que a primera vista nos indican que los sistemas expertos son de gran ayuda, existen por el contrario, un conjunto de problemas o perjuicios con el empleo de los sistemas expertos. El conjunto de limitaciones más destacadas son:

1. El conocimiento es difícil de extraer de los expertos humanos, siendo muchas de las veces escasos de encontrar. Incluso hay a veces que una vez localizado al experto la manera de obtención del conocimiento puede hacerse ardua y muy costosa. La aproximación de un experto a la situación evaluada puede ser diferente a la visión de otros. Además los costos para la extracción en tiempo y dinero suele ser elevado.

2. La capacidad de aprendizaje de los sistemas expertos es muy complicada, pues no resulta sencillo hacerle aprender de los errores que encuentre el mismo o que aprenda de otro sistema en un contexto similar. Por el contrario, un humano experto es capaz de aprender con relativa facilidad de sus errores y de errores ajenos.

3. Actualmente los sistemas expertos depende de una entrada simbólica, mientras que los humanos tienen un amplio rango de disponibilidad de experiencia sensorial.

4. Puede existir una degradación en las respuestas ya que los sistemas expertos no son buenos en el reconocimiento cuando los problemas están fuera de su área o cuando no existe resolución.

5. La programación de los sistemas suele ser difícil de elaborar y precisa de un mantenimiento complejo. Además la poca flexibilidad a cambios de estos sistemas hace que sea necesario una reprogramación.

6. Dificultad a la hora de manipular información no estructurada, especialmente la información incompleta, inconsistente o errónea.

2.1.3. Fuentes de incertidumbre

La incertidumbre es uno de los aspectos que más influyen en los sistemas expertos y en particular en los métodos de razonamiento incierto. Es por tanto necesario identificar que es la incertidumbre y ver cuáles son sus fuentes.

Si nos atenemos a la definición de la ISO (24), define incertidumbre como “una estimación unida al resultado de un ensayo que caracteriza el intervalo de valores dentro de los cuales se afirma que está el valor verdadero”. Es de señalar que esta

definición está en deshuso ya que no tiene una apliación práctica debido a que el "valor verdadero" no puede conocerse.
En nuestro caso se definirá incertidumbre como la falta de información adecuada para tomar una decisión; esto es un problema porque puede evitar que se tome la mejor e incluso puede provocar que se tome una equivocada.

El concepto de incertidumbre, refleja dudas acerca de la veracidad del resultado obtenido una vez que se han evaluado todas las posibles fuentes de error y que se han aplicado las correcciones oportunas. En consecuencia, la incertidumbre nos da una idea de la calidad del resultado ya que mostrará un intervalo alrededor del valor estimado dentro del cual se encontrará el valor considerado como cierto.

A grandes rasgos, se puede clasificar las fuentes de incertidumbre en tres grupos (25), (26):

- Deficiencias de la información, que a su vez se divide en:
    - Incompleta: una información es incompleta respecto a un enunciado si no permite inferir la verdad o falsedad de ese enunciado. Habrá situaciones en las que las limitaciones prácticas, impuestas por el conocimiento del dominio, no permitan contar con todos los medios disponibles para extraer una conclusión.
    - Imprecisa: una información o enunciado es impreciso si admite más de una interpretación posible. Aquellos datos que sean cuantificables son candidatos a información imprecisa ya que son consideradas subjetivamente por distintos expertos.
    - Errónea: aquellos datos susceptibles a ser verdaderos (que la fuente que los suministra no sea viable) y aquellos que puedan dar lugar a falsos positivos y falsos negativos son considerados información errónea.
- Características del mundo real: determinar si las particularidades del conocimiento del dominio se rigen por leyes deterministas o no. No siempre las mismas causas producirán los mismos efectos en el mismo dominio, sin que haya una explicación aparente. Por eso será importante estar abierto a admitir la aleatoriedad y las excepciones del mundo a representar.
- Deficiencias del modelo, que se dividen en:
    - Incompleto: son aquellos conocimientos del dominio o características del modelo que no han sido completamente determinados en todos sus aspectos y recogidos en un conjunto de abstracciones. También estaría incluida toda aquella información que estuviera disponible pero que por motivos prácticos sea imposible incluirla en un sistema experto.

- Inexacto: todo modelo que trate de cuantificar la incertidumbre, por cualquiera de los métodos que existen, necesita incluir un elevado número de parámetros. En el caso de las redes bayesianas será necesario especificar todas las probabilidades *a priori* y condicionales. Sin embargo, esta información no suele estar disponible y es estimada de forma subjetiva, siendo por consiguiente deseable, que el método de razonamiento usado pudiese tener en cuenta las inexactitudes del modelo.

2.1.4. Tipos

Indicar los tipos de sistemas expertos que hay actualmente para la resolución de problemas y clasificarlos dentro del tratamiento de la información a manejar, nos permitirá realizar la elección de la técnica más oportuna dentro del ámbito del problema.

Los problemas que tratan los sistemas expertos pueden clasificarse en dos tipos: problemas fundamentalmente deterministas y problemas principalmente estocásticos. Los primeros son tipos de problemas en los que puede aparecer algo de incertidumbre pero que principalmente contienen información determinista. Como ejemplos se pueden citar sistemas expertos basados en transacciones bancarias y sistemas basados en el control de tráfico. Los segundos, son problemas que también pueden incluir algún elemento deterministas, pero son tratados fundamentalmente como problemas estocásticos debido a la falta de certeza. Siendo de ejemplo, aquellos sistemas basados en el campo médico en las que las relaciones entre los síntomas y las enfermedades se conocen solamente con un cierto grado de certeza.

Los problemas de tipo determinista suelen ser formulados usando conjuntos de reglas que relacionen varios objetos bien definidos. Estos sistemas son conocidos comúnmente como *sistemas basados en reglas*, porque obtienen sus conclusiones a través de un conjunto de reglas que utilizan un mecanismo de razonamiento lógico.

Para el caso en que existan situaciones inciertas, es necesaria la introducción de otros tipos de sistemas expertos que puedan tratar la incertidumbre. Para estos casos de sistemas indecisos, se utilizan algunas fórmulas de propagación para el cálculo de la incertidumbre asociada a las conclusiones. Un caso especial es la inclusión en un sistema basado en reglas de medidas asociadas a la incertidumbre de las reglas y a la de sus premisas, para recoger los contextos aleatorios.

Las propuestas que se han generado en los últimos años en el campo de la IA para medir la incertidumbre son la utilización de técnicas como: *los factores de certeza,* creado por Buchanan y Shortliffe a través del sistema experto MYCIN, cuya base se basa en el concepto de *confirmación* - interpretación lógica de probabilidad (27)); la *lógica difusa* mencionada por primera vez por Zadeh (28)*; la teoría de la evidencia* de A. Demptser y G.Shafer que asocia medidas de incertidumbre a un conjunto de hipótesis (29);y por último la utilización de *la probabilidad,* en la que la distribución conjunta de un acumulo de variables se usa para detallar las relaciones de

dependencia entre ellas, y sacar las conclusiones pertinentes usando las fórmulas de la teoría de la probabilidad.

Los sistemas expertos que utilizan la probabilidad como medida de incertidumbre se conocen como sistemas expertos *probabilísticos* y la estrategia de razonamiento que usa se conoce como *razonamiento probabilístico*, o *inferencia probabilística* (21).

Al inicio del empleo de los sistemas expertos probabilísticos, los mayores problemas vinieron a causa de la dificultad encontrada para definir la distribución de probabilidad conjunta de las variables. Esto ocasionó en cierta medida, la ralentización en el desarrollo de estos modelos. Ha sido posteriormente, con la introducción de los *modelos de redes probabilísticas*, cuando estas dificultades se han superado y la explotación de los sistemas expertos probabilísticos han vuelto a retomar una gran importación en la actualidad.

Estos modelos incluyen en sus conceptos las redes de Markov y las Bayesianas, y se basan en una representación gráfica de las relaciones entre las variables. Esta representación gráfica, es la que permite por un lado, definir de una manera más eficiente las distribuciones conjuntas de probabilidad y por otro, permitir de manera muy eficiente, la propagación de incertidumbre para la obtención de las conclusiones.

El resumen de los tipos de los sistemas expertos, explorados en este apartado, se pueden observar en la figura que se presenta a continuación. En esta evolución se puede dar constancia de la aparición de las distintas técnicas y de cuál es la más acertada para el tipo de información con la que tratan.

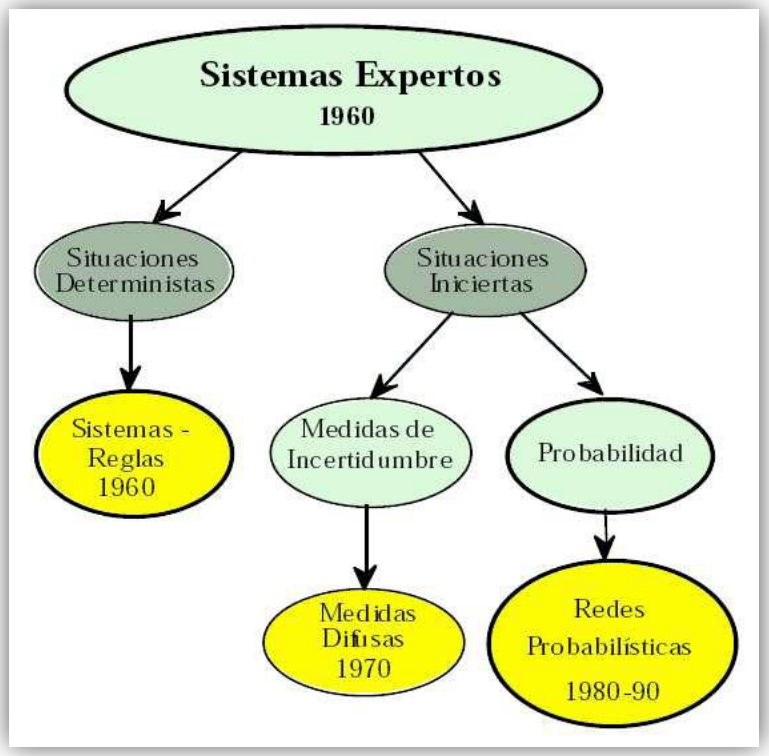

Figura 2.2. Evolución de los sistemas expertos.

2.1.5. Componentes

Para comprender mejor el concepto de un sistema experto, se mostrarán los componentes de estos sistemas. Realmente, no existe una estructura de sistema experto común y como se verá posteriormente, la dimensión de estos puede ser elevada, sin embargo, la mayoría tienen unos componentes básicos. La figura 2.3. Componentes comunes, muestra esta estructura convencional, aunque muchos otros sistemas tienen, además, un módulo de explicación y un módulo de adquisición del conocimiento.

En esta estructura se observa que el motor de inferencia es el elemento central encargado de coordinar todos los demás. Entre ellos, tiene especial importancia la base de conocimiento pues, como su nombre indica, contiene el conocimiento relativo al dominio, ya sea un campo de la medicina, de la ingeniería, etc.

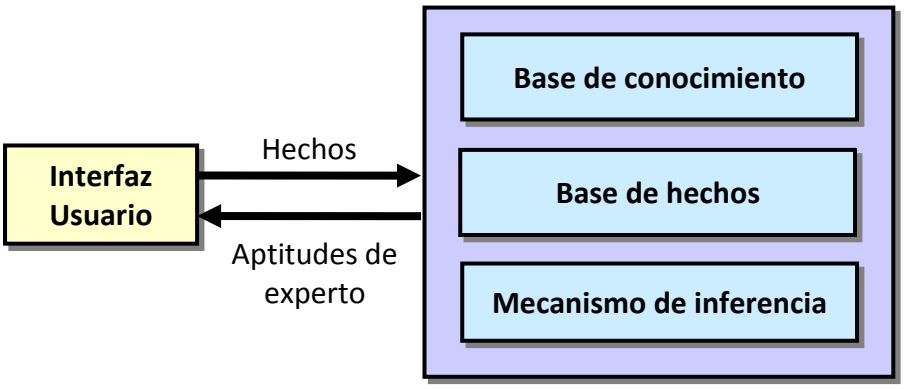

Figura 2.3. Componentes comunes de un sistema experto. Fuente: (18).

A continuación se muestra una breve descripción de cada uno de los componentes de dicha estructura:

1. La base de conocimiento: contiene el conocimiento de los hechos y de las experiencias de los expertos en un dominio determinado extraído del diálogo con el experto.

2. El motor de inferencia: simula la estrategia de solución de un experto imitando el proceso de razonamiento humano.

3. La base de hechos: contiene los hechos sobre un problema que se ha descubierto durante el análisis.

4. La interfaz de usuario: es la interacción entre el sistema experto y el usuario, y que este pueda realizar una consulta en un lenguaje natural.

5. El componente explicativo: este componente puede proporcionar una explicación al usuario de por qué está haciendo una pregunta y cómo ha llegado a una conclusión. Este módulo proporciona beneficios tanto al diseñador del sistema como al usuario.

La estructura presentada anteriormente, es la organización clásica, la cual, se aplica sobre todo a los sistemas basados en reglas. A pesar de ello, en la última década, tras el desarrollo de los distintos tipos de sistemas expertos y el auge de los mismos, la estructura clásica que se presentó, fue insuficiente para recoger los nuevos componentes que fueron surgiendo. Así, se completaron todos aquellos aspectos y se generó un nuevo conjunto de componentes típicos de los sistemas expertos. Esta representación puede observase en la figura 2.4. Componentes desarrollados, donde las flechas representan el flujo de información.

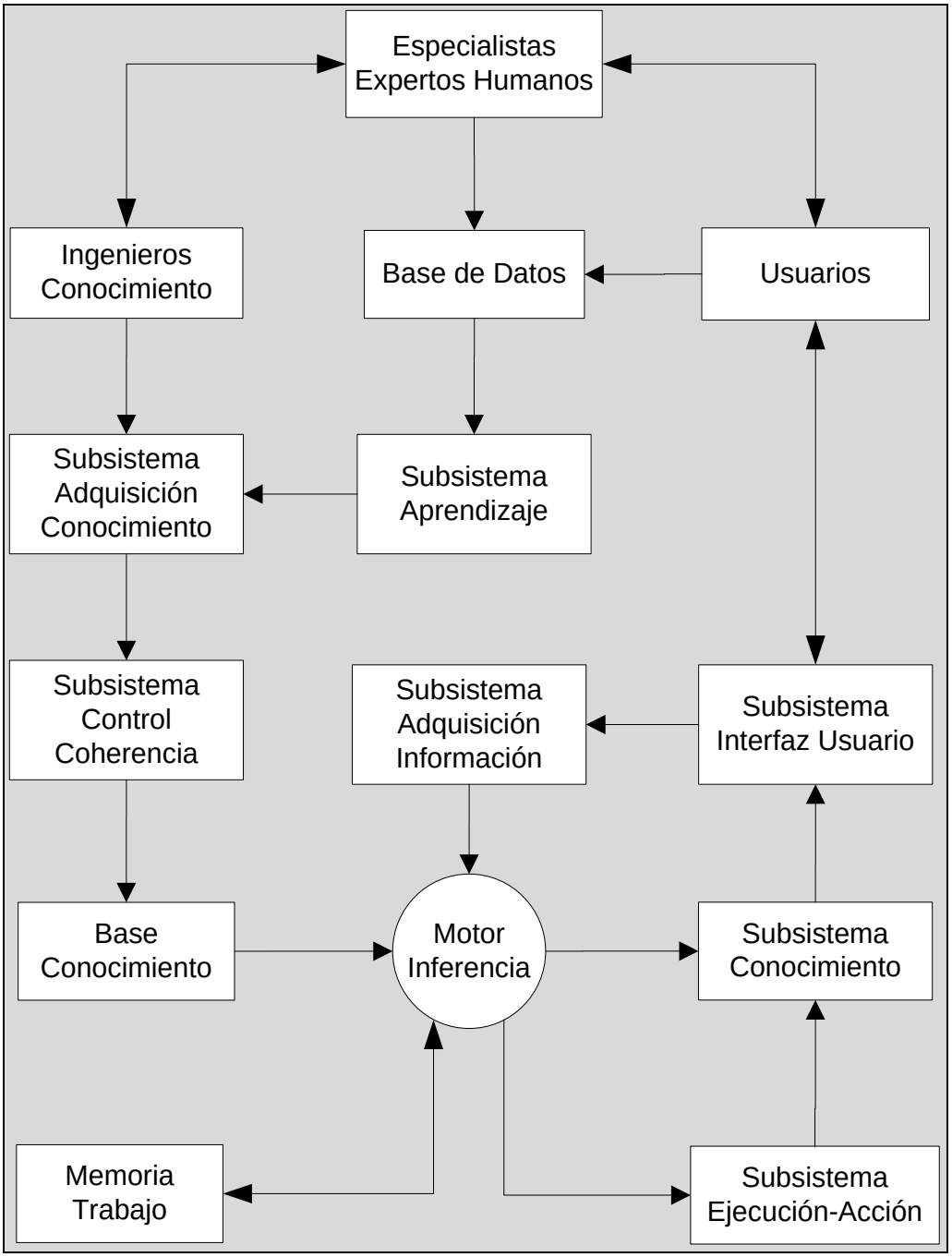

Figura 2.4. Componentes desarrollados de un sistema experto. Fuente: (20).

1. La Componente Humana

En la construcción de un sistema experto es parte esencial la relación entre uno o varios expertos humanos especialistas en el tema de estudio y los ingenieros del conocimiento, ya que de su colaboración se obtendrá un mayor o menor éxito del resultado final del sistema. Por su parte, los expertos se encargarán de facilitar y aportar todo el conocimiento básico sobre el tema de interés, y los ingenieros del conocimiento serán quienes trasladarán este conocimiento de un lenguaje natural, a un lenguaje que el sistema experto pueda tratar. Esta etapa de adquisición, junto con la colaboración de los usuarios del sistema, es el elemento más importante dentro de todo el desarrollo del sistema experto; tanto es así, que esta fase requiere un gran esfuerzo, dedicación y tiempo debido a los diferentes lenguajes que hablan las distintas partes implicadas.

2. La Base del Conocimiento

La base de conocimientos contiene el conocimiento especializado extraído del experto en el dominio a través de diferentes técnicas por parte del ingeniero del conocimiento creando un conjunto de relaciones bien definidas y explicadas. El método más común para representar el conocimiento es mediante reglas de producción, aunque no es el único existente.

Una característica muy importante es, que la base de conocimientos es independiente del mecanismo de inferencia que se utiliza para resolver los problemas. De esta forma, cuando los conocimientos almacenados se han quedado obsoletos, o cuando se dispone de nuevos conocimientos, es relativamente fácil añadir reglas nuevas, eliminar las antiguas o corregir errores en las existentes. Por consiguiente no es necesario reprogramar todo el sistema experto.

El conocimiento se almacena en la base de conocimiento y los datos, información relacionada con una aplicación particular, se almacenan en la memoria de trabajo. Los datos son efímeros y después de usarlos son destruidos por el sistema. A parte de los datos, todo el procedimiento de los distintos sistemas y subsistemas que son temporales se almacenará también en la memoria de trabajo.

3. Subsistema de Adquisición de Conocimiento

El subsistema de adquisición de conocimiento permite que se puedan añadir, eliminar o modificar elementos de conocimiento en el sistema experto en la base de datos. Si el entorno es dinámico, es muy necesario, puesto que, el sistema funcionará correctamente sólo si se mantiene actualizado su conocimiento. El módulo de adquisición permite efectuar ese mantenimiento, anotando en la base de conocimientos los cambios que se producen.

4. Control de la Coherencia

Este subsistema, es parte esencial de un sistema experto a pesar de haber aparecido recientemente en los sistemas expertos. La función del mismo consiste en controlar la

consistencia de la base de datos y evitar que unidades de conocimiento inconsistentes puedan entrar en la misma. Sin este subsistema de control de la coherencia, se podría dar lugar a un comportamiento insatisfactorio del sistema, ya que unidades de conocimiento contradictorio podrían entrar a formar parte de la base de conocimiento.

5. El motor de Inferencia

El motor de inferencia es el centro de todo sistema experto. Este motor de inferencia trabaja con la información contenida en la base de conocimientos y la base de hechos para deducir nuevos hechos. Contrasta los hechos particulares de la base de hechos con el conocimiento contenido en la base de conocimientos para obtener conclusiones acerca del problema.

Las conclusiones del motor de inferencia pueden estar basadas en conocimiento determinista o conocimiento probabilístico. Siendo considerablemente más difícil el tratamiento de situaciones de incertidumbre (probabilísticas) que el tratamiento de situaciones ciertas (deterministas) (21).

En el caso de tener un conocimiento incierto en el sistema experto, será además función del motor de inferencia, propagar la incertidumbre.

6. El Subsistema de Adquisición Información

En el caso en que el conocimiento inicial sea muy limitado y no se pueda obtener conclusiones, es cuando el motor de inferencia necesitará el subsistema de adquisición de información para poder adquirir un mayor conocimiento y continuar con el proceso de inferencia para obtener las conclusiones. Esta información necesaria es suministrada por usuario a través de la interfaz de usuario.

7. Interfaz de Usuario

La interacción entre el sistema experto y los usuarios se realiza a través de la interfaz de usuario usando un lenguaje lo más natural posible. Esta interfaz debe de mostrar la información fácilmente, eficientemente y con una calidad agradable para los usuarios. La interfaz, permitirá al usuario poder describir el problema al sistema experto cuando el motor de inferencia no pueda obtener una conclusión debido a la ausencia de información. Al ser la interfaz de usuario el enlace entre el sistema experto y el usuario, es importante que este sea de una calidad notable para que facilite el proceso de comunicación y no lleve a prejuzgamientos erróneos de la eficacia del sistema experto.

8. El Subsistema de Ejecución de Órdenes

En caso de que el sistema experto tenga que realizar determinadas acciones a partir de las conclusiones sacadas por el motor de inferencia, será necesaria la inclusión de un subsistema de ejecución de órdenes. Al ser este subsistema el encargado de realizar las acciones oportunas, del sistema experto por el que fue diseñado, dará lugar al subsistema de explicación que explique las razones del porqué de estas acciones.

9. El Subsistema de Explicación

Como ya he mencionado, en el subsistema de ejecución de órdenes, el subsistema de explicación será el encargado de mostrar las conclusiones obtenidas por el motor de inferencia que explique todo el proceso seguido hasta la salida final obtenida. Sin este sistema pueden darse situaciones en las que la explicación de las conclusiones no se obtenga debido a que no se puede seguir los pasos realizados por el motor de inferencia hasta la obtención del resultado.

10. El Subsistema de Aprendizaje

Este es el subsistema que dota de una de las principales características de los sistemas expertos, la capacidad de aprendizaje. En este subsistema se pueden hacer dos distinciones sobre el tipo de aprendizaje. Por un lado se tiene el aprendizaje estructural y por otro el aprendizaje paramétrico.

El aprendizaje estructural se refiere a los aspectos relacionados con la estructura del conocimiento (reglas, distribuciones de probabilidad, etc.). Y el aprendizaje paramétrico se refiere a la estimación de los parámetros necesarios para la construcción de la base de conocimiento.

2.1.6. Desarrollo

Como hemos podido comprobar en los apartados anteriores, un sistema experto es un sistema basado en conocimiento que tiene su dominio en la inteligencia artificial, siendo importante para su desarrollo y construcción del modelo basado en conocimiento, los mecanismos para la adquisición de conocimiento. Ésta es considerada como la tarea más complicada en la ingeniería del conocimiento y, posteriormente su representación correspondiente.

Es de tener en cuenta que en el desarrollo del sistema experto se tendrá que usar la Ingeniería del Software, que permite desarrollar el proyecto aportando herramientas de software, técnicas, procedimientos-estructurales, aspectos técnicos, aspectos financieros, etc. Para poder iniciar y llegar a la obtención del sistema final de software existen una serie de etapas o procedimiento que se deben de llevar a cabo y que se denominan *Ciclo de Vida*, este ciclo consiste en la concepción, construcción, implementación y la aplicación del producto, abarcando desde el comienzo del concepto inicial del software hasta que termina con la retirada de su uso.

Basados en este ciclo, se han establecidos paradigmas o modelos para optimizar el desarrollo del software, cada modelo tiene sus ventajas y desventajas, siendo la selección del modelo, dependiente de la magnitud del proyecto de desarrollo, de los costos y del tiempo que desea invertirse.

El diseño e implementación de un sistema experto ha sido dividido en distintas etapas para una mejor comprensión de los pasos a seguir para su desarrollo. En la figura 2.5 Fases de desarrollo de un sistema experto, se observan el resumen de las etapas sugerido en 1984 por Weiss y Kulikowski (30). Las flechas indican la secuencia a seguir

entre las distintas etapas. En la figura es de destacar las posibles reformulaciones o vueltas atrás entre etapas cuando la finalización de la misma no es la adecuada o, cuando se considera que no se cumplen los objetivos mínimos deseados.

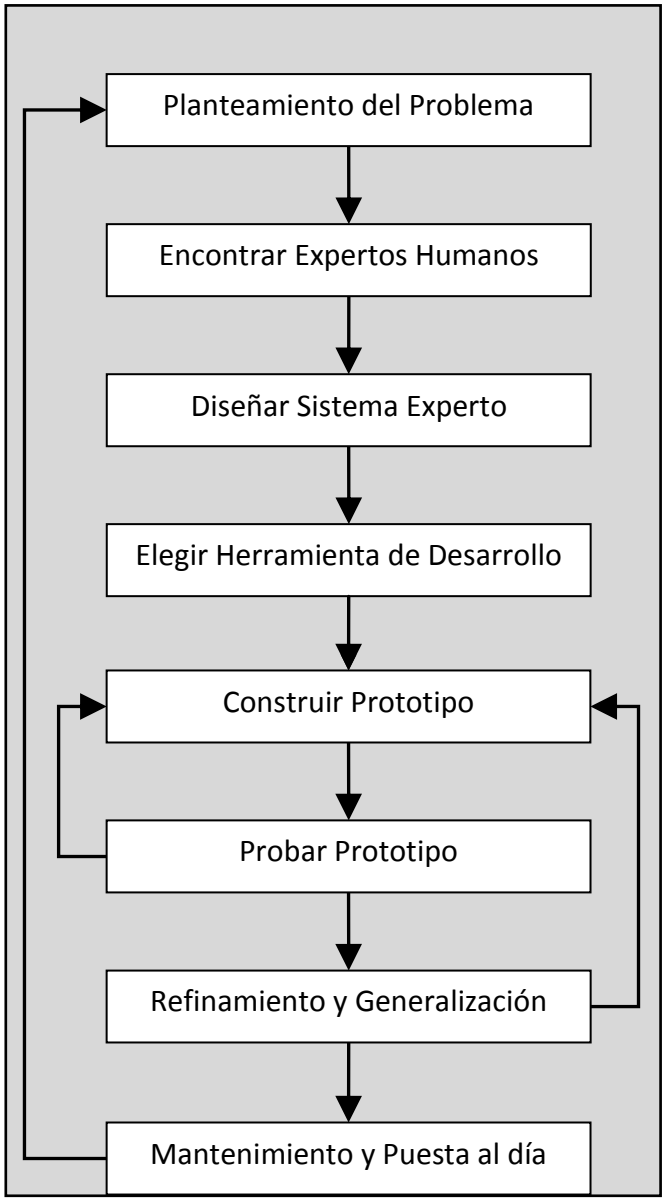

Figura 2.5. Fases de desarrollo de un sistema experto. Fuente: (30)

La descripción de las etapas que influyen en la calidad del sistema experto son:

1. **Planteamiento del problema.** La primera fase del proyecto consiste en la definición del problema. Se debe de describir el problema en términos concretos, explícitos y específicos, de manera que todos los aspectos dudosos queden parcial o totalmente acotados para su posterior resolución. Al ser la primera etapa de todas, quizás sea la más importante ya que un problema correctamente planteado está parcialmente resuelto y las posibilidades de éxito del sistema experto son mayores a la finalización del mismo.

2. **Encontrar expertos humanos que puedan resolver el problema.** Planteado el problema y acotado el ámbito de trabajo a resolver, es recomendable la extracción del conocimiento con ayuda de un experto, que aporte su conocimiento y experiencia. En algunos casos, la utilización de las bases de datos pueden utilizarse como sustitución del experto humano.

3. **Diseño de un sistema experto.** En esta etapa, se realiza el diseño de las estructuras necesarias para almacenar la abstracción realizada en la etapa anterior. Es en este momento cuando se realiza también el diseño de todos los componentes del sistema experto.

4. **Elección de la herramienta de desarrollo, o lenguaje de programación.** Vistos en el apartado 2.1.4 los distintos tipos de sistemas expertos, y elegido uno, es necesario elegir una herramienta, o un lenguaje de programación que mejor se ajuste a las necesidades. En caso de existir herramientas que satisfagan los requerimientos del diseño, se optará por una de ellas ya que existirá un beneficio tanto en la fiabilidad como en los recursos económicos del sistema experto.

5. **Desarrollo y prueba de un prototipo**. En esta etapa, se desarrollará un prototipo acorde al diseño del sistema experto. Tras su finalización se realizarán las pruebas requeridas, realizando las modificaciones apropiadas en etapas anteriores, hasta que el prototipo sea satisfactorio.

6. **Refinamiento y generalización.** Durante esta etapa, se perfecciona el prototipo desarrollado en la fase anterior, se corrigen los fallos y en caso de detectarse nuevas posibilidades no incorporadas en el diseño inicial, se retrocede y se añaden volviendo a realizar la secuencia de pasos.

7. **Mantenimiento y puesta al día.** Tras la finalización del proyecto, queda solamente plantear posibles problemas o defectos del prototipo que hayan surgido posteriormente, corregir dichos errores y actualizar el producto con nuevas mejoras.

Capítulo III. Redes bayesianas

Para el desarrollo del caso práctico del proyecto es necesario disponer del conocimiento necesario de la técnica de redes bayesianas. En el presente capítulo, se mostrarán aquellos aspectos más importantes que se han considerado necesarios para la construcción, resolución y entendimiento del proyecto. Ciertos detalles que no están recogidos en esta memoria, pueden ser consultados en la inmensa bibliografía existente sobre las redes bayesianas. Especialmente, se recomienda, ver (31) para una mayor información, ya que es un libro íntegramente dedicado a todos los aspectos relacionados con las redes bayesianas.

El formalismo de las redes bayesianas facilitará la representación eficiente y el razonamiento riguroso en situaciones en las que se dispone de conocimiento incierto. Actualmente, es una de las técnicas que dominan la investigación de la inteligencia artificial en el razonamiento incierto y en los sistemas expertos. Esta aproximación facilita el aprendizaje a partir de la experiencia, y combina lo mejor de la inteligencia artificial clásica y las redes neuronales (32).

3.1. Definición formal

Como se vio en el apartado 2.1.4 Tipos, las redes bayesianas son una alternativa a la hora de implementar un sistema experto probabilístico ya que posen cierta cualidades que otras técnicas no permiten, como por ejemplo, admiten el aprendizaje sobre relaciones de dependencia y causalidad, permiten la combinación de conocimiento con datos, evitan el sobre-ajuste continuo de los datos y pueden manejar bases de datos incompletas.

A parte del nombre de red bayesiana (el más común), existen muchas otras denominaciones con las que se pueden referirse también a las redes bayesianas y sobre las que se refieren también distintos autores como son redes de creencia, redes probabilística, redes causal y mapas de conocimiento.

Formalmente, una red bayesiana es un es un grafo acíclico dirigido (*DAG*) en la cual cada nodo representa una variable de un dominio y cada arco una dependencia probabilística, en la cual se especifica la probabilidad condicional de cada variable dados sus padres.

La red Bayesiana se puede ver como un conjunto *(V, D, P)* donde:

- *V* es el conjunto de variables del dominio que se quiere representar.
- *D* es el grafo acíclico dirigido (*DAG*) cuyos nodos están etiquetados con los elementos de V. Siendo los arcos dirigidos los indicadores de la relación de influencia y en algunos casos relación causal.
- *P* es la distribución conjunto sobre las variables (*V*).

La variable a la que apunta el arco es dependiente (causa-efecto) de la que está en el origen de éste. Y la topología o estructura de la red nos dará información sobre las dependencias probabilísticas entre las variables. A parte de la topología será necesario contar con las distribuciones de probabilidad *a priori* de los nodos padres y de la probabilidad condicionada de los nodos hijos, que aporte la información cuantitativa.

Una red bayesiana tiene al menos un nodo raíz (un nodo sin padre alguno) y un nodo terminal (un nodo sin hijos) (4).

La figura 3.1., muestra un ejemplo de red bayesiana, que determina si la hierba de un determinado lugar esta mojada o no, dependiendo de unas variables aleatorias que describen el problema.

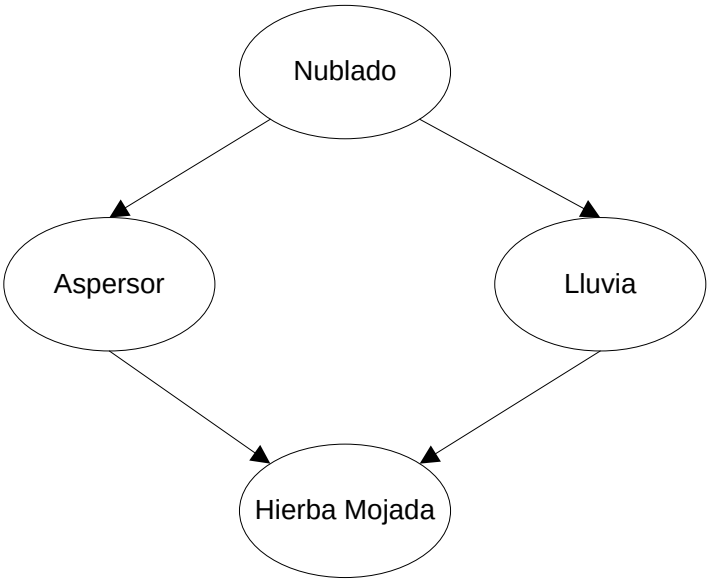

Figura 3.1. Ejemplo simple de red bayesiana. Adaptado de (33).

En esta red, cada nodo se corresponde con una variable, que a su vez representa una entidad del mundo real. Los arcos que unen los nodos indican las relaciones de influencia causal entre dichas variables. Para que la figura estuviese completa faltarían las probabilidades que se asocien a cada variable que se recogen en tablas llamadas tablas de probabilidad condicionada (TPC).

3.1.1. Separación direccional

Las redes Bayesianas, nos permiten disponer de una serie de hipótesis que facilitan el cálculo de los valores de las probabilidades, debido a la independencia condicional y/o separación direccional (llamada *d-separation* por Pearl), que es equivalente a la propiedad de Markov, que afirma que un nodo es independiente de los nodos que no son descendientes suyos dados sus padres.

Dado un grafo dirigido acíclico conexo y una distribución de probabilidad sobre sus variables, se dice que hay separación direccional si dado un nodo X, el conjunto de sus padres separa condicionalmente este nodo de todo otro subconjunto Y en el que no haya descendientes de X. Es decir:

$$P(x \mid Padres\ (x), Y) = P(x \mid Padres\ (x))$$

Por ejemplo, en la red de la figura 3.1., vista anteriormente, la propiedad de separación direccional permite deducir, por ejemplo, que la variable *Hierba Mojada* es independiente de la variable *Nublado,* conocidos los valores de *Aspersor* y *Lluvia*.

Es tan importante la propiedad de separación direccional, que como consecuencia, se tiene que la probabilidad conjunta de una red bayesiana se puede obtener como el producto de las probabilidad de cada nodo condicionadas a sus padres, lo que se conoce como el **teorema de factorización de la probabilidad**:

$$P(x_1, \ldots, x_n) = \prod_{i=1..n} P(x_i \mid Padres(x_i))$$

3.1.2. Tipos de redes bayesianas

Existen varios tipos de redes bayesianas dependiendo del tipo de variables que contenga el grado (discretas, continuas, o ambas) y del tipo de distribución que se considere para cada variable (21).Si las variables del problema son todas discretas, el modelo asociado es una *red bayesiana discreta* o *red bayesiana multinomial*. Si las variables del problema siguen una distribución normal, la red es una *red bayesiana normal* o *red bayesiana Gaussiana* y aunque se trabaja en la definición de una red bayesiana para otro tipo de variables aleatorias continuas, todavía no se ha determinado el mecanismo de inferencia cuando la red está formada por otro tipo de variables continuas no Gaussianas (34). La inclusión de ambos tipos de variables en la red darán lugar a las *redes bayesianas Mixtas*. Además de las anteriores redes las cuales son dependientes del tipo de variables, si el razonamiento probabilístico está condicionado por el tiempo, existirá un cuarto tipo de red denominada *red bayesiana Dinámica* (que se mostrarán posteriormente más detalladamente en el apartado 3.6., debido a que el desarrollo del entorno de usuario a problemas de fusión de información requerirá disponer de un conocimiento más detallado de este tipo de redes).

Antes de continuar con la explicación más detallada de cada uno de los distintos tipos de redes bayesianas, mencionadas anteriormente, es necesario hacer un inciso sobre los modelos gráficos probabilísticos no dirigidos, denominadas también redes de Markov, como un tipo especial de modelo gráfico que no puede ser considerado como una red bayesiana, pero que poseen ciertos aspectos comunes.

En las redes de Markov, la información cualitativa del problema es representada por un grafo no dirigido, donde las relaciones de dependencia entre las variables del problema son relaciones de asociación o correlación, sin establecerse ninguna variable como causa o como efecto, de forma que la información de la que se dispone indica que un conjunto de variables presenta distintos niveles de asociación o correlación (34).

Posteriormente a la construcción del grafo no dirigido, se buscará la distribución de probabilidad conjunta asociada a las variables del problema como una factorización de funciones. Los usos más habituales de la redes de Markov están orientados en campos como la física, la robótica y en análisis de imágenes y de textos.

La principal deficiencia de los grafos no dirigidos es su incapacidad para representar relaciones de independencia no transitivas; en estos modelos, dos variables independientes estarán conectadas en el grafo siempre que exista alguna otra variable que dependa de ambas (21). Por tanto, numerosos modelos de dependencia útiles desde un punto de vista práctico no pueden ser representados por grafos no dirigidos, y es pues, donde aparece la necesidad de representar las relaciones causales a través de los grafos dirigidos y, más concretamente, con el uso de las redes bayesianas.

3.1.2.1. Redes Bayesianas Discretas

Estas redes se caracterizan porque todas las variables que aparecen en el modelo son discretas, de manera que cada variable sólo puede tomar un conjunto determinado de valores. Cuando las variables del problema son binarias, respondiendo a los procesos de Bernoulli, la red se denomina como red bayesiana Multinomial.

La figura 3.2., presenta un ejemplo de red bayesiana discreta, en el que además de presentar el diagrama de relaciones se muestran las distribuciones de probabilidad condicionada asociadas a los valores de las variables, donde cada variable tiene determinado sus valores. Los valores posibles de cada variable serán la apreciación de que se produzcan o no, tras su observación, siendo las respuestas sí o no.

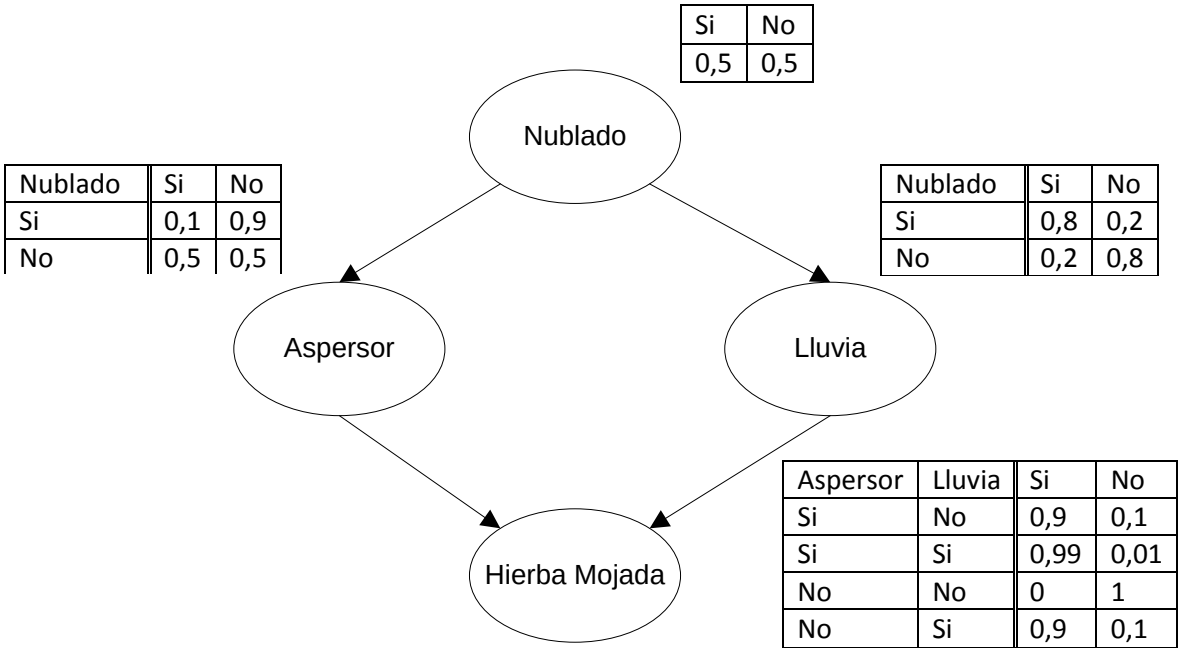

Figura 3.2. Ejemplo de red bayesiana discreta.

3.1.2.2. Redes Bayesianas Gaussianas

En las redes bayesianas Gaussianas, las variables aleatorias del problema son normales, siendo la distribución conjunta de las variables del problema X = {X_1, ….,X_n} una normal multivariante N(μ,Σ), dada por la función de densidad

$$f(x) = (2\pi)^{-n/2}|\Sigma|^{-1/2} exp\{-1/2(x-\mu)^T\Sigma^{-1}(x-\mu)\}$$

donde μ es el vector de medias de dimensión n, Σ es la matriz de covarianzas de tamaño n x n, |Σ| es el determinante de Σ, y $(x-\mu)^T$ es el vector traspuesto de (x –μ).

Partiendo del teorema de factorización de la probabilidad, y de la densidad de distribución normal de las variables se obtiene que

$$f(x_i|pa(X_i)) \sim N\left(\mu_i + \sum_{j=1}^{i-1}\beta_{ij}(x_j-\mu_j), \upsilon_i\right)$$

donde β_{ij} es el coeficiente de regresión de X_j en la regresión de X_i sobre sus padres, y υ_i es la varianza condicionada de X_i dados sus padres, siendo $\upsilon_i = \Sigma_i - \Sigma_{ipa(X_i)}\Sigma^{-1}_{pa(X_i)}\Sigma^T_{ipad(X_i)}$, siendo $\beta_{ij} = 0$ si y solo si no hay una arista dirigida de X_j a X_i.

Para la obtención de una mayor información sobre este tipo de redes bayesianas y ampliar los conocimientos de los mismos se recomienda consultar (34).

3.1.2.3. Redes Bayesianas Mixtas

Este tipo de redes se caracterizan por utilizar tanto variables discretas como variables continuas en el modelo gráfico probabilístico dirigido.

Al combinar los dos tipos de variables y poderlas especificar en el modelo, las variables continuas han de ser Gaussianas y las variables discretas deberán preceder a las continuas en el grado.

En estas redes el conjunto de nodos V = {X_1,…, X_n}, formada por ambos tipos de variables, se divide por un lado en discretas (Δ) y por el otro en continuas (Γ), siendo V = Δ ∪ Γ. Así, el conjunto de todas las variables aleatorias se denotan como,

$$X = (x)_{\propto\in V} = (i,\varsigma) = ((i_\delta)_{\delta\in\Delta}, (\varsigma_\gamma)_{\gamma\in\Gamma})$$

siendo la distribución condicionada Gaussiana, cuya densidad vienen dada por

$$f(x) = f(i,\varsigma) = \exp\{g(i) + h(i)^T\varsigma - \varsigma^T K(i)\,\varsigma/2\}$$

donde i representa las variables discretas y ς las continuas, *g(i)* es un escalar, *h(i)* un vector, *K(i)* una matriz definida positiva y $h(i)^T$ denota el vector *h(i)* traspuesto.
Una mayor información sobre las redes bayesianas mixtas pueden consultarse en (35).

3.2. Construcción

Para la construcción de una red bayesiana, es necesario realizar varias tareas hasta conseguir una estructura final dispuesta a funcionar dentro del sistema experto. Actualmente existen dos formas de construir una red bayesianas, que se pueden denominar como *automática* y *manual* (36); también se podría incluir como otro proceso de construcción la combinación de ambos tipos.

Ambas formas, aunque de distinta manera, implican en su proceso de construcción básicamente tres tareas:

1. Identificar las variables y sus valores.

2. Identificar las relaciones entre las variables, completando la definición del grafo que representa el modelo.

3. Obtener las probabilidades asociadas a cada nodo del grafo.

- El proceso *manual*, construye la red bayesiana a partir de la ayuda de un experto humano que conozca a fondo el problema que se quiere modelar. Así a través de esta ayuda, el ingeniero del conocimiento, establecerá primero la estructura de la red causal (fase cualitativa), y posteriormente añadirá las probabilidades condicionales (fase cuantitativa) de los nodos creados. La construcción manual de la red bayesiana será la estructura central del diseño de la aplicación desarrollada, por eso se detallarán en profundidad los pasos posteriormente.

- El proceso *automático* consiste en tomar una base de datos en la que todas las variables que nos interesas estén representadas y que contenga un número de casos suficientemente grande. Aplicando entonces alguno de los algoritmos que se han desarrollado recientemente para esta tarea, se obtienen los enlaces y las probabilidades condicionales que definen la red bayesiana.
  Sin embargo, en muchos problemas reales, es muy difícil contar con una base de datos suficientemente grande y detallada para la construcción de la red. De la construcción automática se hablará brevemente de los algoritmos que se pueden aplicar para dar una ligera idea de las posibilidades en el desarrollo, ya que estos algoritmos se apoyan en un desarrollo matemático bastante complejo para poder contemplarlos en el presente proyecto.

- La combinación de ambas posibilidades, permite orientar al experto y al ingeniero del conocimiento para afianzar o corregir su percepción del dominio. Se puede optar por obtener el modelo de forma manual, a través de la ayuda de expertos humanos y aplicar alguno de los algoritmos de aprendizaje para la obtención de las probabilidades. Por otro lado, también se puede aprender la red a partir de una base de datos y posteriormente realizar una depuración refinando la estructura y los parámetros con la ayuda de expertos humanos.

3.2.1. Construcción manual

La construcción manual de modelos gráficos probabilísticos no es una tarea trivial y no existen unos criterios definidos que se puedan aplicar constantemente ante cualquier problema. Aún así, en esta tarea se cuenta normalmente con la experiencia previa del experto humano, si existiese en su caso, para aventurar las relaciones entre las variables; pero normalmente el sentido común y la experiencia propia del ingeniero del conocimiento serán los aspectos más importantes a la hora de la construcción manual de una red bayesiana.

Los elementos implicados en la construcción manual de redes bayesianas así como las distintas fases necesarias que se describirán a continuación están basados en la estructura presentada en la Tesis Doctoral, Explicación en redes bayesianas causales. Aplicaciones Médicas, de Carmen Lacave (15).

3.2.1.1. Elementos y fases

En la construcción de un manual de diagrama de influencia o de una red bayesianas es importante recopilar todas las fuentes de información relacionadas con el dominio a modelar, para obtener así el mayor conocimiento posible. Los aspectos más importantes que se tienen que realizar son:

1. Obtener bibliografía sobre el dominio a trabajar, que puede obtenerse de distintos medios como son libros, actas de conferencias, revistas, medios electrónicos, informes, etc.

2. Disponer de una herramienta de edición y procesado de redes bayesianas que recoja las necesidades del desarrollo, con el fin de ayudar al ingeniero de conocimiento tanto en la construcción como en la depuración de la red.

3. Contar en la medida de los posible, con la colaboración de al menos un experto humano, ya que será el único modo de obtener la información ausente y corregir la errónea. Aunque no es necesario la colaboración del experto humano, sí que es recomendable ya que la bibliografía sobre el domino puede estar desactualizada, ser imprecisa o no contener los datos que uno necesita. Como ya se comentó anteriormente en la segunda fase de desarrollo del sistema experto en el apartado 2.1.6. Desarrollo, la utilización de una base de datos suficientemente amplia y completa del dominio, puede utilizarse como sustitución del experto humano.

Tras la adquisición de los elementos anteriores, el proceso de construcción de la red bayesiana consta básicamente, de dos partes:

- **Fase cualitativa:** consiste en la definición del grafo que representará las variables, previamente seleccionadas, del problema y las relaciones de dependencia e independencia entre ellas. El primer paso consistirá en la identificación de las variables del modelo y así poder empezar con un modelo inicial que sirva de

referencia para que, en etapas posteriores y mediante refinamientos sucesivos, se vaya aproximando al diseño de la red definitiva.
Un apartado importante será determinar cuántos valores tendrá cada variable, pues en un principio, cuanto más detallado sea el modelo, mayor precisión tendrá. Sin embargo, la construcción de modelos más completos conlleva, por un lado, la obtención de un conjunto mayor de probabilidades numéricas (con lo que se pierde fiabilidad en los datos obtenidos) y, por otro, que la computación consuma un mayor tiempo de respuesta, además de que la interpretación de los resultados sea más difícil.
Esta etapa requiere una notable cantidad de tiempo ya que existen evidencias empíricas (37) en las que se señala que es más importante para el correcto funcionamiento del sistema la estructura de la red que la precisión que se pueda obtener de los datos numéricos.

- **Fase cuantitativa:** consiste en la obtención de los datos cuantitativos, es decir, de las probabilidades condicionadas de las variables que presenten nodos padres y de las probabilidades *a priori* para aquellas variables que son nodos raíz. La tarea de adquisición de las probabilidades es la labor más complicada de la construcción manual de una red bayesiana. En principio, lo correcto sería que todos los datos procedieran de estudios estadísticos para obtener de manera sencilla las probabilidades buscadas. La realidad muestra que es casi imposible disponer exactamente de los datos que uno necesita sobre todo si la obtención de las probabilidades no se hace de forma automática a partir de bases de datos y no siempre existirán registros con la información deseada. Para estas situaciones, surgieron los *modelos canónicos*, que permiten construir tablas de probabilidad con mucho valores a partir de un pequeño conjunto de parámetros, aportando un ahorro tanto de espacio de almacenamiento como de tiempo de propagación. Los modelos canónicos representa relaciones entre un nodo y sus padres y no son exclusivos de las redes bayesianas, sino que se usan también en diagramas de influencia y pueden aplicarse a las redes de Markov (38). Los principales tipos de modelos canónicos son:

  - Modelo de interacción disyuntiva o puerta OR (*Noisy OR-gate*).
  - Modelo de interacción conjuntiva o puerta AND (*Noisy AND-gate*).
  - Puerta MAX (*Noisy Max- gate*).
  - Puerta MIN (*Noisy Min-gate*).

  Para una mayor información sobre los conceptos de los modelos canónicos se pueden consultar las siguientes referencias bibliográficas, para la puerta OR (1) y para el resto de las puertas, MIN, MAX y AND (39).

La realización de estas fases es un proceso continuo y no tiene porque ser independiente ya que una vez se está en una fase no se puede dar por concluidas las anteriores. El proceso en sí de construcción de una red bayesiana se asemeja a un proceso iterativo, ya que según se va avanzando se puede obtener o modificar la información de las etapas anteriores realizando por tanto un diseño descendente con posibles refinamientos hasta la obtención de un grafo con sus probabilidades lo más

adecuadas posibles. Por consiguiente en el proceso de construcción de la red se podrá ir obteniendo distintas versiones de la red y así, explorar las exactitudes de cada una de ellas.

Un número demasiado grande de variables puede ser un problema tanto en la construcción de la red como en su evaluación y aplicación. Esto puede ocasionar que resulte imposible obtener la precisión teórica deseada debido a la complejidad de la obtención de las probabilidades que ello conlleva. Por el contrario, cuanto menor sea el número de variables mayor será el riesgo de que los resultados obtenidos sean poco reales. En teoría los resultados serán más preciso cuanto mayor sea el número de variables. Por tanto, habrá que llegar a una solución de compromiso para mantener un equilibrio entre el número de variables y la eficiencia de la red.

La identificación de los modelos canónicos, cuando se puedan utilizar, puede ayudar en la obtención de probabilidades y así simplificar dicho proceso tan complicado en la construcción manual de la red.

Finalmente, si el modelo resultante de todas las fases es aceptado y no requiere de refinamientos en los datos numéricos ni de nuevas variables y/o enlace, se terminará el proceso de construcción y ya se dispondrá de una red dispuesta para el funcionamiento dentro del sistema experto.

Todo el proceso y las etapas seguidas para la construcción del proceso manual de una red bayesiana, puede observarse en la figura 3.3. en la página siguiente.

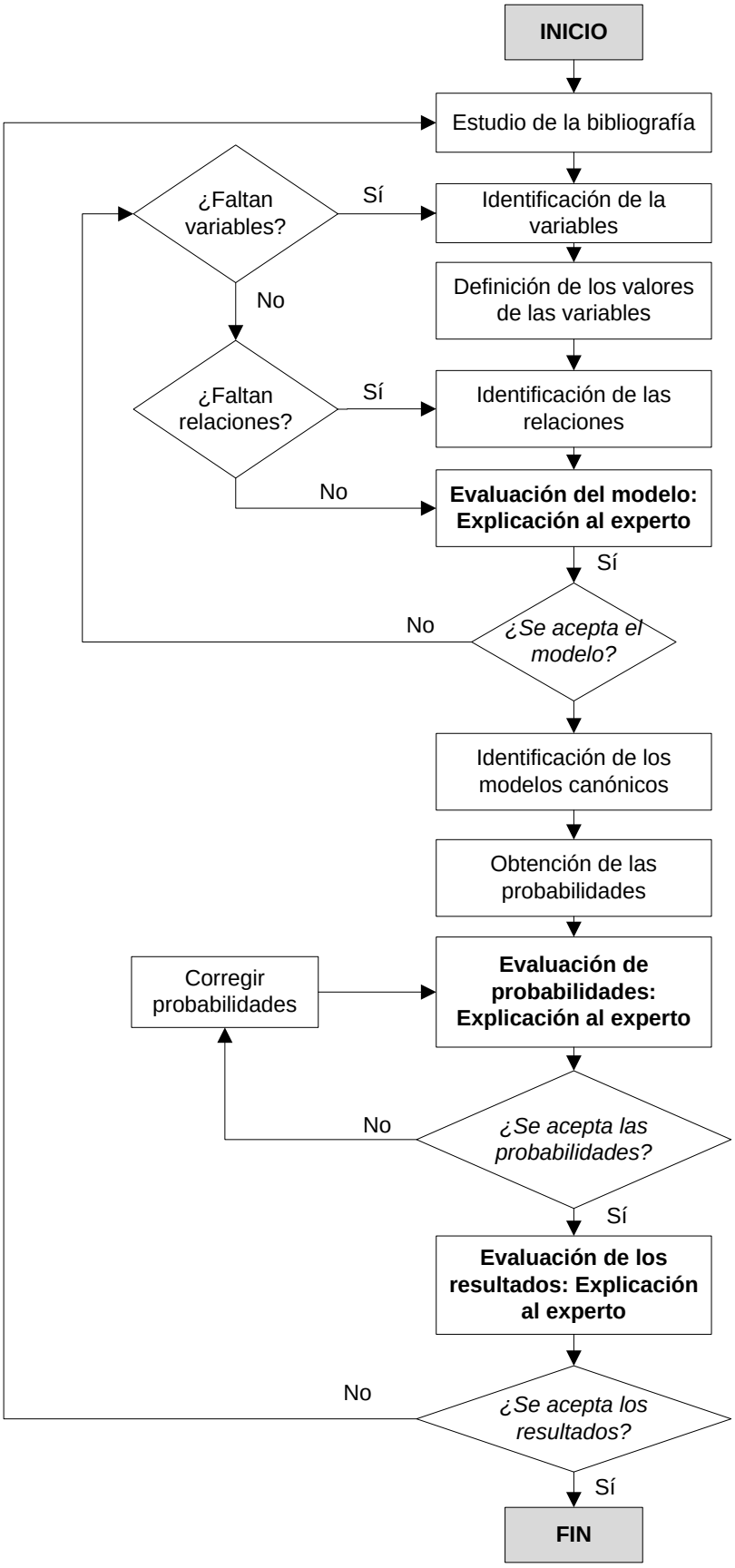

Figura 3.3. Algoritmo para la construcción manual de redes bayesianas.

3.2.2. Construcción automática

El conjunto de técnicas que se emplean para la construcción de las redes bayesianas de manera *automática*, son conocidas como *algoritmos de aprendizaje*, que permitirán extraer toda la información necesaria.

Según Pearl (1) existen dos fases de aprendizaje, cuando se trabajan con redes bayesianas, que se puede denominar respectivamente aprendizaje estructural y aprendizaje paramétrico. Estas dos fases se pueden resumir como:

- ***Aprendizaje paramétrico:*** a partir de la estructura de la red, se obtiene las probabilidades *a priori* de los nodos raíz y las probabilidades condicionales de las demás variables requeridas a través del uso de bases de datos.

- ***Aprendizaje estructural:*** obtener la estructura de la red bayesiana a partir de bases de datos, obteniendo las relaciones de dependencia e independencia entre las variables existentes. Los algoritmos que aprenden la estructura de la red bayesiana se engloban generalmente dentro de una de las categorías siguientes (40).

    - Algoritmos que se basan en un procedimiento que busca la mejor estructura en el espacio de posibles soluciones, midiendo la calidad de cada red candidata mediante funciones de evaluación. Estos son algoritmos que se caracterizan por el tipo de función y por el procedimiento de búsqueda.

    - Algoritmos basados en detección de independencias, que toman como entrada el conjunto de relaciones de independencia condicional y generan la red que mejor representan estas relaciones.

    - Algoritmos híbridos que se basan en la combinación de ambas metodologías.

En el aprendizaje de redes bayesianas, será casi requisito principal para poder realizar la tarea de aprendizaje a partir de datos, disponer de bases de datos muy extensas en las que estén especificado el valor de cada variable en cada uno de los casos.

En el apartado 3.4. Aprendizaje de clasificadores bayesianas y en el apartado 3.5. Aprendizaje de redes bayesianas, se mostrarán algunos de los distintos tipos de algoritmos de clasificación más conocidos que más se usan actualmente y las técnicas o métodos necesarios para el aprendizaje tanto paramétrico como estructural en las redes bayesianas.

3.3. Inferencia

Una vez construida la estructura de la red y halladas las distribuciones de probabilidad para cada nodo, el siguiente paso es determinar el cambio producido en estas probabilidades cuando los valores de algunas de las variables llegan a ser conocidos. A este proceso de instanciación de variables de entrada y propagación de sus efectos a través de la red, es lo que se llama propagación de probabilidades.

La propagación de probabilidades está relacionada al razonamiento probabilístico y consiste en propagar los efectos de la evidencias a través de la red para conocer la probabilidad a posteriori de las variables. A pesar de la utilización de la independencia condicionada de las variables para simplificar el proceso de propagación de la evidencia, la propagación en redes bayesianas es un problema NP-Completo. Esto es debido a la estructura gráfica de la red bayesiana que a pesar de ser un diagrama acíclico dirigido, en la propagación de la evidencia se modifica dicha estructura gráfica pudiendo existir ciclos no dirigidos, que hacen intratable el proceso de propagación de la evidencia. No obstante en la mayoría de los casos se llega a una solución mediante un proceso eficiente.

Existen diferentes tipos de algoritmo para calcular las probabilidades posteriores. Los principales tipos de propagación que existen, dependientes del tipo de grafo son:

- Algoritmos de propagación en árboles.
- Algoritmos de propagación en poliárboles.
- Algoritmos de propagación en redes multiconectadas.

3.3.1. Propagación en árboles

El algoritmo que se aplica en las estructuras de tipo árbol se puede extender fácilmente a los poliárboles, pero no se puede aplicar en las redes multiconectadas.

Dada cierta evidencia *E*, representada por la instanciación de ciertas variables, la probabilidad a posterior de cualquier variables *B*, por el teorema de Bayes es

$$P(B_i|E) = P(E|B_i)P(B_i)/P(E)$$

Al ser la estructura de la red un árbol, el nodo *B* la separa en dos subárboles (ver figura 3.4., en la página siguiente), dividiendo la evidencia en dos grupos:

- *E-*: datos en el árbol cuya raíz es el nodo *B*.
- *E+*: datos del resto del árbol.

Donde $E = E^- \cup E^+$

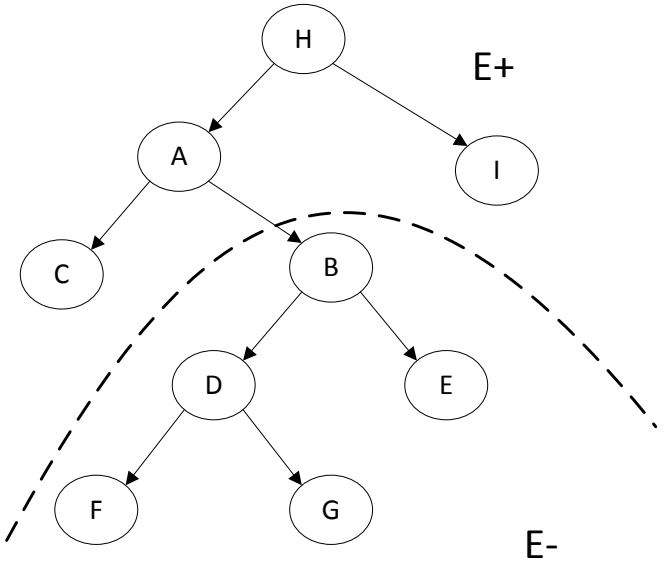

Figura 3.4. Propagación en un árbol.

Que sustituyendo en la anterior ecuación da como resultado:

$$P(B_i|E) = P(E-, E+|B_i)P(B_i)/P(E)$$

Pero, dado que ambos son independientes y aplicando nuevamente el teorema de Bayes:

$$P(B_i|E) = \alpha P(B_i|E+)P(B_i|E-)$$

Donde α es la constante de normalización. Y así se consigue separar la evidencia para la actualización de la probabilidad de *B*. Además de no ser necesario requerir a la probabilidad *a priori*, excepto en el caso de la raíz donde:

$$P(A_i|E+) = P(A_i)$$

Si se definen los siguientes términos como:

$$\lambda(B_i) = P(E-|B_i)$$
$$\pi(B_i) = P(B_i|E+)$$

Entonces por sustitución:

$$P(B_i|E) = \alpha\pi(B_i)\lambda(B_i)$$

Una vez obtenida la anterior ecuación, la obtención de la probabilidad para un nodo dada una cierta evidencia, se puede integrar en un algoritmo distribuido. Para ello, se divide el proceso en dos partes: por un lado la evidencia de los hijos (λ), y por otro la evidencia de los demás nodos (π). Donde cada nodo almacenará los valores de los vectores λ y π, junto a las matrices de probabilidad de *P*. El proceso de propagación se realiza por medio de un mecanismo de paso de mensajes, en donde cada nodo envía los mensajes correspondientes a sus padres y a sus hijos.
Mensaje al padre (hacia arriba), del nodo *B* a su padre *A*:

$$\lambda_B(A_i) = \sum_j P(B_j|A_i)\,\lambda(B_j)$$

Mensaje a los hijos (hacia abajo), nodo *B* a su hijo S_k:

$$\pi_k(B_i) = \alpha\pi(B_j)\prod_{l\neq k}\lambda_l(B_j)$$

Al instanciarse nuevos nodos, éstos enviarán mensajes a sus padres e hijos, y se propagarán hasta llegar a la raíz u las hojas, o hasta encontrar un nodo instanciado. Al finalizar el proceso de propagación, cada nodo tendrá un vector π y un vector λ. Y se obtendrá la tabla de probabilidad multiplicando término por término de acuerdo a la ecuación:

$$P(B_i|E) = \alpha\pi(B_i)\lambda(B_i)$$

La propagación se realiza una sola vez en cada sentido y en un tiempo proporcional al diámetro de la red.

3.3.2. Propagación en poliárboles

El poliárbol, es uno de los modelos gráficos más simples para construir redes bayesianas. El algoritmo de propagación es muy similar al de árboles, visto anteriormente. La principal diferencia es que se requiere de la probabilidad conjunta de cada nodo dado todos sus padres. Si en el caso de la figura 3.5. se supone que ahora el nodo *B* posee $A_1,\ldots,A_n$, nodos padres, entonces se requerirá:

$$P(B_i|A_i, \ldots A_n)$$

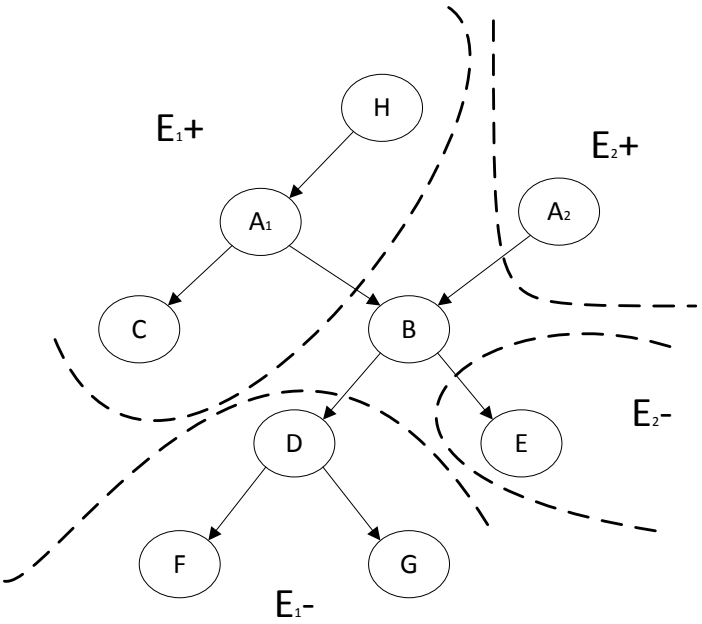

Figura 3.5. Divisiones en partes del algoritmo de propagación en poliárboles.

De donde se puede deducir la expresión de la probabilidad en un nodo cualquiera *B* en términos de sus padres e hijos, tal que:

$$P(B_i|E) = \alpha(P(B_i|E_1+, \dots E_n +)P(E_1 - |B_i) \dots P(E_m - |B_i)$$

El algoritmo de propagación para este tipo de modelos probabilísticos más utilizado es paso de mensajes diseñado por Kim y Pearls (41). La característica principal de este algoritmo es que su complejidad es lineal en el tamaño de la red, a diferencia de otros métodos de fuerza bruta que requieren un número exponencial de operaciones para realizar la propagación. Sin embargo, la principal limitación del algoritmo de Kim y Pearls es que no permite tratar los bucles que aparecen inevitablemente al desarrollar modelos, de redes bayesianas del mundo real, por lo que su utilización resulta de poca utilidad. Debido a esto, los diseñadores de aplicaciones de redes bayesianas recurren a otros algoritmos que, aun perdiendo las ventajas de este, son aplicables a todo los tipos de estructuras de redes bayesianas.

Para una descripción detallada del algoritmo puede ser consultada en el libro *Bayesian Artificial Intelligence* (31).

3.3.3. Propagación en redes multiconectadas

Tanto el método de propagación en poliárboles como el método de propagación en árboles, son válidos para estructuras simples en los cuales solamente existe un único camino entre cada par de nodos. Al ser este tipo de redes tan simples, la mayoría de veces carecen de generalidad y no pueden ser aplicables en numerosas situaciones prácticas, por lo que es necesario, en estos casos, trabajar con grafos múltiplemente conexos (grafos que contienen bucles) en los que pueden existir varios caminos entre dos nodos.

Los métodos de propagación que se pueden utilizar en este tipo de redes se pueden agrupar en:

- **Método de condicionamiento:** se basan en bloquear las trayectorias de propagación al instanciar una variable. Después se asumen valores para un grupo seleccionado de variables para poder descomponer la red en un conjunto de poliárboles. Por último se realizará la propagación para cada valor posible de dichas variables usando el algoritmo de paso de mensajes para poliárboles.

- **Método de simulación estocástica:** consiste en asignar valores aleatorios a las variables no instancias, calcular la distribución de probabilidad y obtener valores de cada variable dando una muestra. Este procedimiento se repite para obtener un número apreciable de muestras y en base al número de ocurrencias de cada valor se determinará la probabilidad de dicha variable. La técnica más utilizada dentro de la simulación estocástica es el método de Monte Carlo también denominado como método de filtro de partículas.

- **Método de agrupamiento:** este método también conocido como árbol de uniones (*junction tree*), consiste en transformar la estructura de la red para obtener un árbol, mediante agrupación de nodos usando la teoría de grafos.

Para ello, se parte de la red original y tras sucesivas transformaciones se pasa a un árbol de uniones (grupo de nodos) mediante el siguiente procedimiento:

1. Se elimina la direccionalidad de los arcos.
2. Se ordenan los nodos por máxima cardinalidad.
3. Moralizar el grafo (arco entre nodos con hijos comunes).
4. Se triangulariza el grado agregando los arcos adicionales necesarios.
5. Se identifican todos los conjuntos de nodos totalmente conectados (*cliques*).
6. Se ordenan los *cliques* de forma que todos los nodos comunes estén en un solo *clique* anterior (su padre).
7. Se construye un nuevo grafo en que cada *clique* es un nodo formando un árbol de *cliques*.

Por último, la propagación de probabilidades se realiza en este árbol formado del conjunto de *cliques*, obteniendo la probabilidad conjunta de cada *clique*, donde a partir de ésta se puede obtener la probabilidad individual de cada variable en el *clique*.

La figura 3.6. muestra la transformación seguida para el caso de una red sencilla.

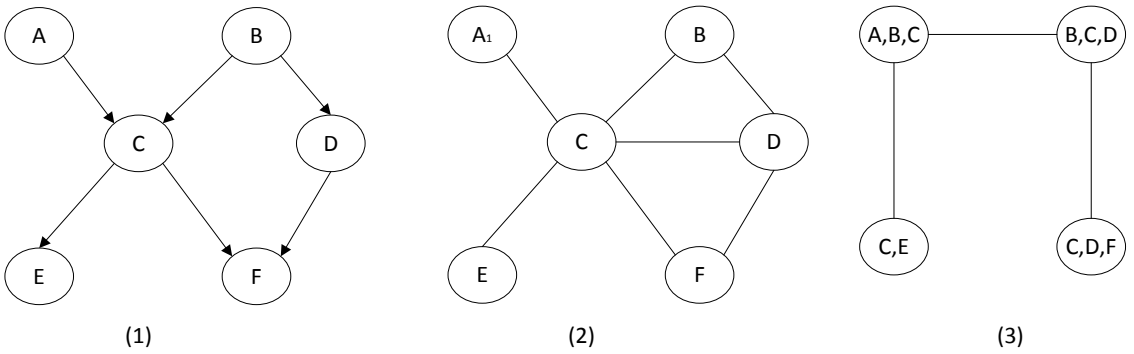

Figura 3.6. Transformación de una red a un árbol de uniones: (1) red original, (2) red moralizada y triangulada, (3) árbol de uniones.

El problema de propagación de evidencias en redes bayesianas múltiplemente conexas es un problema NP-Complejo (21). Por lo general, los tres métodos anteriores, tienen éste problema de complejidad. Sin embargo, debido a las características particulares de cada método, la utilización de uno u otro método de propagación hacen que sea más eficiente en redes con estructuras particulares. No obstante, en general, ninguno de los tres métodos es más eficiente que el otro, sino que suelen ser complementarios, lo que ha ocasionado la aparición de nuevos algoritmos mixtos que combinan las ventajas de ambos métodos.

En el caso de tener redes muy complejas, en las que existan muchas conexiones, la mejor alternativa posible es el empleo de técnicas de simulación estocásticas.

3.4. Aprendizaje de clasificadores bayesianos

Las redes bayesianas están diseñadas para encontrar las relaciones de dependencia e independencia entre todas las variables que conforman el dominio de estudio. Esto permite realizar predicciones sobre el comportamiento de cualquiera de las variables desconocidas a partir de los valores de las otras variables conocidas. Permitiendo que cualquier variable de la base de datos pueda comportarse como incógnita o como evidencia según sea el caso.

Siendo la tarea de clasificación como un caso particular de la tarea de predicción, mencionada anteriormente, de las redes bayesianas.

Un clasificador puede verse como un caso especial de una red bayesiana en el cuál una función asigna un valor de un atributo discreto, llamado clase, a instancias o conjuntos de características especificados por atributos, que pueden ser tanto continuos como discretos. En el caso de que la base de conocimiento sea una red bayesiana, la función de clasificación estará definida a partir de las probabilidades condicionadas. Siendo la estructura de esta red dependiente del tipo de clasificador que se emplee, como se verá posteriormente.
Los clasificadores bayesianas tienen un uso extendido debido a que presentan ciertas ventajas:

1. Por lo general, son fáciles de construir y de entender.
2. El proceso de inducción suele ser considerablemente rápido, requiriendo solamente uno o varios pasos.
3. Son muy robustos cuando existen atributos irrelevantes.
4. Toman evidencias de muchos atributos para realizar la predicción.

A continuación se explican los clasificadores bayesianos más importantes.

3.4.1. Naïve Bayes (NB)

El clasificador Naïve Bayes es una de las redes bayesianas más eficientes en el proceso de clasificación, que a pesar de su simplicidad es comparable con clasificadores sofisticados como las redes neuronales y los árboles de decisión, ya que posee una alta precisión y velocidad cuando es aplicada a un conjunto grande de datos.

La estructura de este clasificador puede observarse gráficamente en la figura 3.7. (ver página siguiente), en la que existe un nodo para la variable de clase, que es el padre de todas las demás variables y en la que no se permiten arcos entre las variables, asumiendo la restricción de que los atributos son independientes conocido el valor de la variable clase. Teniendo por consiguiente, una estructura de red fija donde solamente se necesitará aprender los parámetros (probabilidades).

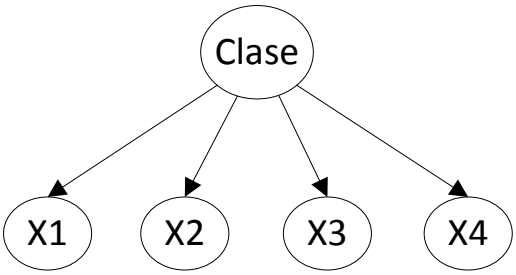

Figura 3.7. Ejemplo de estructura de un clasificador Naïve Bayes.

Como un ejemplo de problema en el que el clasificador NB se está mostrando como una de las técnicas más eficaces, se puede citar la lucha contra el correo basura o *spam* Muchos lectores de correo incorporan este clasificador para etiquetar el correo no solicitado (42).

3.4.2. Tree Argumented Naïve Bayes (TAN)

El clasificador TAN, es también conocido como clasificador bayesiano simple aumentado como un árbol.

Este método de clasificación es una extensión del clasificador NB. Se basa principalmente en construir una red bayesiana más compleja que admita la conexión entre nodos en forma de árbol, asumiendo que el conjunto de atributos pueden ser causalmente dependientes. De tal manera, que en principio se tendrán pocas conexiones entre los atributos, con lo que no se amentará demasiado la complejidad de la estructura, pero donde se intentará mejorar la tasa de acierto del proceso de clasificación.

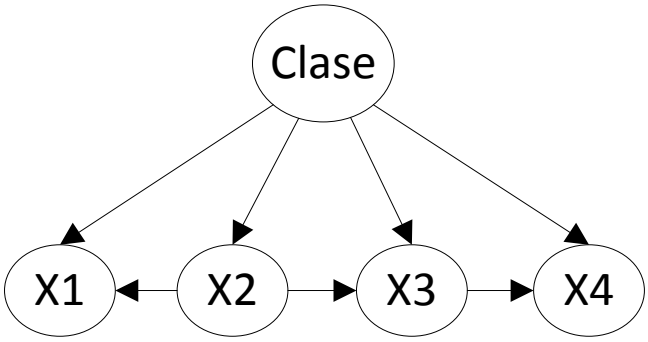

Figura 3.8. Ejemplo de estructura de un clasificador TAN.

Actualmente existen distintas variaciones de este algoritmo, entre los que se pueden citar:

- Métodos de aprendizaje de redes bayesianas que parten de la estructura del NB como estado inicial y, sucesivamente van añadiendo arcos hasta llegar a una estructura de TAN.

- Métodos que simultáneamente al proceso de construcción realizan una selección de variables (sTAN).

- Métodos que distribuyen los atributos en un acumulo de árboles, en lugar de un único árboles (FAN, de *Forest Augmented Naïve Bayes*)

3.4.3. Bayesian Network Augmented Naïve Bayes (BAN)

El algoritmo BAN, se basa en la filosofía de TAN. La principal diferencia de este clasificador bayesiano simple, es que adopta una estructura de red. En el clasificador BAN se procede aprendiendo una red bayesiana para los atributos (excluyendo la clase) y posteriormente aumentando el modelo añadiendo la variable de clase y aristas desde éste hacia todos los atributos.

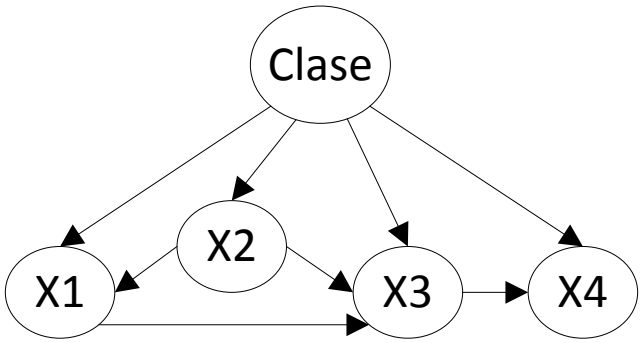

Figura 3.9. Ejemplo de estructura de un clasificador BAN.

3.4.4. Structured Augmented Naïve Bayesian Network (SAN)

El algoritmo SAN (en español red bayesiana estructurada simple aumentada) es más flexible que el algoritmo TAN, en el sentido de que permite la construcción de estructuras bayesianas simples aumentadas menos restrictivas (43). Estas estructuras, se caracterizan porque la clase no tiene padres y los atributos de entrada pueden tener como padres además de la clase, cualquier número de atributos de entrada, siempre que no haya ciclos dirigidos, ya que la estructura que únicamente permite este algoritmo son grafos acíclicos dirigidos.

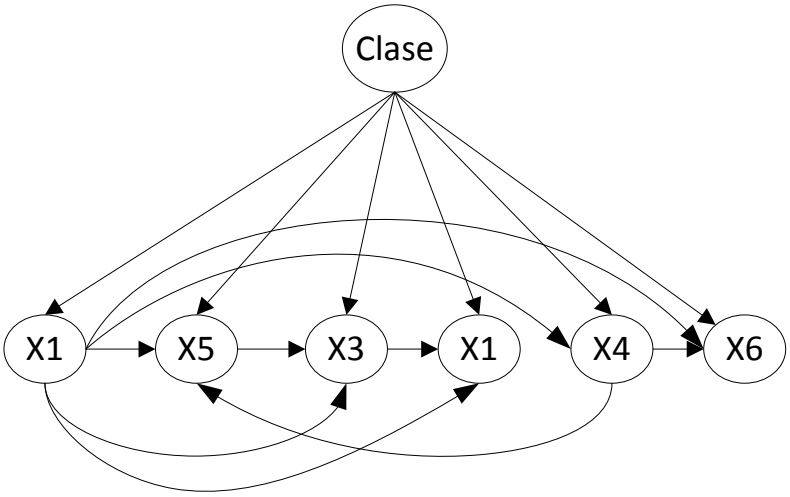

Figura 3.10. Ejemplo de estructura de un clasificador SAN.

No obstante, cuantos más arcos existan en una estructura SAN, mayor es el riesgo de que la estructura aprendida clasifique bien los casos usados para el aprendizaje, pero ocasionando que haya una baja eficiencia para casos nuevos, haciendo que su capacidad de generalización sea baja y por tanto el aprendizaje no se pueda considerar aceptable.

3.4.5. Clasificador bayesiano K-Dependiente

El algoritmo *K-Dependence Bayesian classier (kDB)* presentado por Sahami (44), posibilita atravesar el amplio espectro de dependencias disponibles entre el modelo Naïve Bayes y el modelo correspondiente a una red bayesiana completa.
Su algoritmo se basa en contener la estructura del clasificadro NB y permitir a cada variables predictiva tener un máximo de k variables padres sin contar a la variable clase. De tal modo que el modelo Naïve Bayes se corresponda con un clasificador Bayesiana 0-dependiente, el modelo TAN un clasificador bayesiana 1-dependiente y el clasificador bayesiana completo (en el que no exista ninguna independencia) se corresponderá a un clasificador bayesiana (n-1)-dependiente.

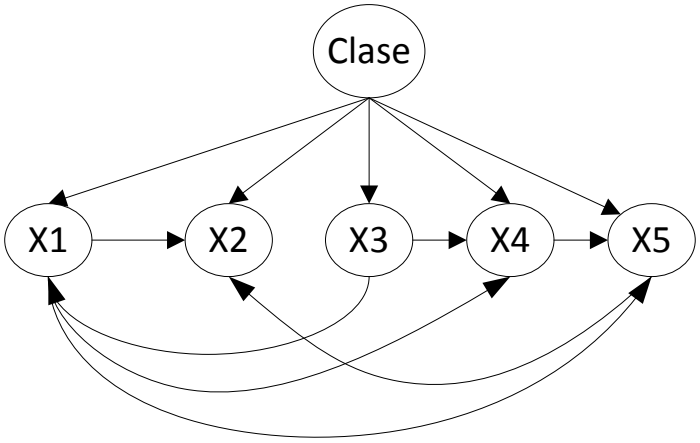

Figura 3.11. Ejemplo de estructura en la construcción de kDB con k =2.

3.5. Aprendizaje de redes bayesianas

Durante mucho tiempo la construcción de las redes bayesianas se ha realizado a mano a partir del conocimiento adquirido de los expertos. Actualmente, existen distintos tipos de algoritmos, métodos y/o técnicas que permitirán obtener tanto la estructura como los parámetros de la red.

En las redes bayesianas el aprendizaje consiste en definir la red probabilística a partir de datos contenidos en bases de datos en lugar de obtener el conocimiento a través del experto humano. Con este tipo de aprendizaje, existe la posibilidad de conseguir la estructura gráfica de la red a partir de los datos observados y poder definir las relaciones entre los nodos basándose también en dichos casos. El aprendizaje por lo general es una tarea muy compleja. A partir del incremento del número de variables (nodos), el número de posibles grafos a construir se eleva considerablemente. Por eso en muchas ocasiones se restringe el espacio de búsqueda a grafos con características concretas.

De manera informal se describirá el problema de aprendizaje bayesiana como: dado un conjunto de entrenamiento **E** = {$\mathbf{e_1}$, $\mathbf{e_2}$,..., $\mathbf{e_n}$} de instancias de **E**, encontrar la red que se ajuste mejor a **E**.

3.5.1. Aprendizaje paramétrico

Como ya se mencionó en el aparatado 3.2.2 Construcción automática, el aprendizaje paramétrico tiene como objetico encontrar los parámetros asociados a una estructura dada de una red bayesiana. En este proceso de aprendizaje, se tienen que hacer dos distinciones en base al conocimiento de los datos, por un lado si los datos están incompletos y por otro si están completos, ya que uno u otro tipo de datos, determinarán la manera de obtención de los parámetros.

3.5.1.1. Datos completos

En el caso de conocer todas las variables en el conjunto de entrenamiento, el proceso de obtención de las probabilidades requeridas serán fáciles de obtener. Las probabilidades *a priori* corresponden a las marginales de los nodos raíz y las condicionales se obtienen de las conjuntas de cada nodo con sus padres.

El método más común en este caso es el llamado *estimador de máxima verisimilitud (*del inglés *ML, maximun likelihood)* que consiste sencillamente estimar las probabilidades deseadas a partir de la frecuencia de los valores de los datos de entrenamiento, de manera análoga al proceso seguido en el clasificador Naïve Bayes. Para una red bayesiana se tienen dos casos:

- Nodos raíz: se estima la probabilidad marginal. Por ejemplo, $P(A_i) \sim nA_i/n$ donde nA_i es el número de ocurrencias del valor *i* de la variable A, y *n* es el número total de casos.

- Nodos hoja: se estima la probabilidad condicional de las variables dados su(s) padre(s). Por ejemplo, $P(B_i|A_j, C_k) \sim nB_iA_jC_k/n\,A_jC_k$, donde $nB_iA_jC_k$ es el número de casos en que $B = B_i, A = A_j\ y\ C = C_k$ y nA_jC_k es el número de casos en que $A = A_j\ y\ C = C_k$.

Dependiendo del número datos, la calidad de estas estimaciones variará. Si éstos, son insuficientes, será necesario cuantificar la incertidumbre existente representándola mediante una distribución de probabilidad, para así considerarla explícitamente en la definición de las probabilidades (45). Si las variables son binarias se empleará una modelización con una distribución Beta y en el caso de variables, se empleará su extensión, que es la distribución Dirichlet.

3.5.1.2. Datos incompletos

En la mayoría de los casos, los datos no están completos en las bases de datos con las que uno dispone. En estos casos el aprendizaje en redes bayesianas consiste en inferir de alguna manera los datos ausentes para completar la base de datos.

Se pueden distinguir dos tipos básicos de información incompleta:

- ***Valores faltantes:*** faltan valores de una de las variables en algunos casos.

- ***Nodos ocultos:*** faltan todos los valores de una variable.

Para el primer tipo de información incompleta, el de los valores faltantes, las posibles alternativas para el tratamiento de los datos en este caso son:

- Eliminar los casos (registros en la base de datos) donde aparecen valores faltantes.
- Considerar un nuevo valor adicional para la variable, como 'desconocido.'
- Tomar el valor más probable (la media aritmética) de la variable.
- Considerar el valor más probable en base a las otras variables.
- Considerar la probabilidad de los diferentes valores en base a las otras variables.

En los dos primeros casos, suelen ser adecuadas cuando se cuente con un número elevado de datos, ya que si no se desaprovecha información al eliminar o sustituir valores. La tercera alternativa, no suele ocasionar los mejores resultados ya que no considera las posibles dependencias de la variable con las demás. Por último, el cuarto y el quinto caso son por lo general los más importantes por generar los mejores resultados.

Para el caso del valor más probable se puede aplicar el algoritmo presentado en la tabla 3.1., es cuál es la forma más simple de realizar el aprendizaje de los nodos escondidos.

Tabla 3.1. Algoritmo sencillo para el descubrimiento de datos incompletos (46).

1.	Instanciar todas las variables observables.
2.	Propagar su efecto y obtener las probabilidades posteriores de las no observables.
3.	Para las variables no observables, asumir el valor con probabilidad mayor como observado.
4.	Actualizar las probabilidades previas y condicionales de acuerdo a las formulas anteriores.
5.	Repetir (1) y (4) para cada observación.

En el caso de la existencia de nodos ocultos, el tratamiento para el cálculo de las tablas de probabilidad condicional es bastante más complejo que las alternativas presentadas para los valores faltantes. El método más comúnmente aplicado es el empleo del algoritmo de *EM (del inglés Expectation Maximization)*.

El algoritmo EM es un método estadístico muy utilizado en la estimación de probabilidades cuando hay variables no observables. Este algoritmo recibe su nombre de los dos pasos en los que se basa cada iteración:

- Paso E: se estiman los datos faltantes en base a los parámetros actuales.

- Paso M: se calculan los valores de los parámetros considerandos los datos estimados.

En el caso de nodos ocultos en redes bayesianas, el algoritmo EM se puede consultar en la tabla 3.2. y se resume como:

Tabla 3.2. Algoritmo EM en redes bayesianas.

1. Iniciar los parámetros desconocidos (probabilidades condicionales) de la red, con valores aleatorios (o basados en el conocimiento de expertos si se disponen de él).
2. Emplear los datos conocidos con los parámetros actuales para estimar los valores de la variable(s) ocultas(s).
3. Utilizar los valores calculados para completar la tabla de datos.
4. Volver a recalcular los parámetros con los nuevos datos.
5. Repetir los pasos 2 a 4 hasta que no haya cambios significativos en las probabilidades.

La principal limitación del algoritmo EM es que puede ocasionar máximos locales, por lo tanto, los valores finales obtenidos pueden depender de los parámetros de inicialización.

3.5.2. Aprendizaje estructural

En el apartado 3.4. Aprendizaje de clasificadores bayesianos, se vieron una serie de clasificadores, que eran un caso particular de una red bayesiana en la tarea de predicción.

Como ya se definió, el aprendizaje estructural permite encontrar las relaciones de dependencia entre las variables, de tal forma que se puede determinar la topología o estructura de la red bayesiana. Por tanto, en esta sección se explicarán los distintos algoritmos que permiten realizar el aprendizaje estructural de acuerdo a los tres tipos de estructuras posibles:

- Aprendizaje de árboles.
- Aprendizaje de poliárboles.
- Aprendizaje de redes multiconectadas.

En primer lugar, se explicará el método de aprendizaje de árboles para posteriormente mostrar su extensión a la estructura de poliárboles. Por último se mostrará el aprendizaje de las redes multiconectadas con los dos tipos de enfoques que se pueden aplicar.

3.5.2.1. Aprendizaje de árboles

El aprendizaje estructural de árboles se basa en el algoritmo desarrollado por Chow y Liu (47) para aproximar una distribución de probabilidad por un producto de probabilidades de segundo orden (correspondiente a un árbol). La probabilidad conjunta de *n* variables se puede presentar como:

$$P(X_1, X_2, \dots, X_n) = \prod_{i=1}^{n} P(X_i | X_{j(i)})$$

donde $X_{j(i)}$ es el padre de X_i.

Para conseguir el árbol que nos relación las variables, el problema de aprender la estructura de la red bayesiana a través de los datos se plantea como uno de optimización, donde se quiere obtener la estructura de árbol que más se aproxime a la distribución 'real'. Por lo que será necesario, basarse en una medida de diferencia de información entre la distribución real (*P*) y la aproximada (*P**), siendo la diferencia:

$$DI(P, P^*) = \sum_X P(X) \log(P\,(X)/P^*(X))$$

Donde el objetivo será minimizar DI, pudiéndose definir dicha diferencia en función de la información mutua entre pares de variables, que se define como:

$$DI(X_i, X_j) = \sum_{X_i, X_j} P(X_i, X_j) \log(P\,(X_i, X_j)/P(X_i)P(X_j))$$

Donde se puede demostrar que la diferencia de información es una función del negativo de la suma de las informaciones mutuas (pesos) de todos los pares de variables que constituyen el árbol. Siendo el objetivo, encontrar la estructura de árbol más próximo equivalente a encontrar el árbol con mayor peso (47).

Basándose en estos conceptos, se puede encontrar el árbol óptimo mediante el algoritmo de la tabla 3.3., que es equivalente al conocido problema del "*maximun weight spanning tree"*. Donde el único problema es que el algoritmo no provee la direccionalidad de los arcos, por lo que la asignación de la dirección podrá asignar de forma arbitraria o a través de la adquisición del conocimiento del experto.

Tabla 3.3. Algoritmo de obtención de árbol óptimo.

1. Calcular la información mutua entre todos los pared de variables (que en el caso de *n* variables, son *n(n-1/2)*).
2. Ordenar las informaciones mutua de mayor a menor.
3. Seleccionar la rama que mayor valor tenga como árbol inicial.
4. Añadir la siguiente rama mientras no se forme un ciclo. En caso de que se forme habrá que desecharla.
5. Repetir el paso 4 hasta que se cubran todas las variables (*n-1* ramas).

3.5.2.2. Aprendizaje de poliárboles

Para el aprendizaje de poliárboles se emplea el algoritmo de Rebane y Pearl (48), el cuál es a su vez una extensión del algoritmo seguido por Chow y Liu para el aprendizaje estructural en árboles[3], basándose en dar direcciones a la estructura mediante pruebas de independencia no sólo entre dos variables, sino entre grupos de tres variables o tripletas.

[3] Hay que recordar que un árbol es un caso especial de poliárbol.

El algoritmo de Rebane y Pearl parte del esqueleto (estructura sin direcciones) obtenido a través del algoritmo de Chow y Liu. Y después, se determinan las direcciones de los arcos probando las relaciones de dependencia entre todas las tripletas de variables existentes en el esqueleto. Estas tripletas de variables pueden formar tres tipos de arcos:

1. Arcos convergentes. X → Y ← Z.
2. Arcos divergentes. X ← Y → Z.
3. Arcos secuenciales. X → Y → Z.

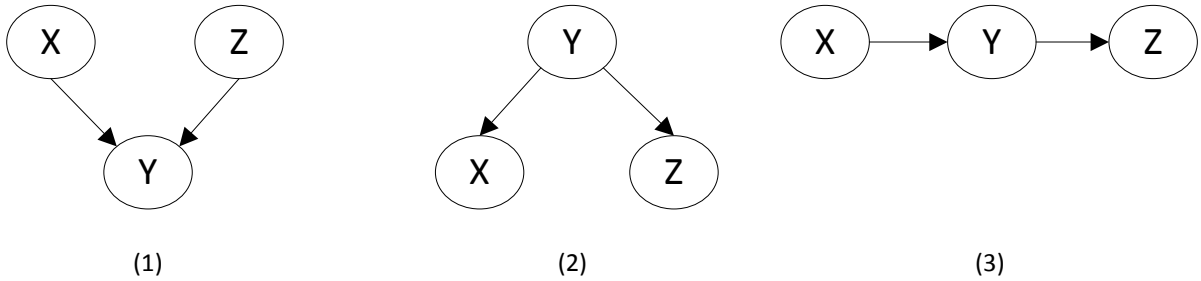

Figura 3.12. Tipos de conexiones en un grafo dirigido. (1) Convergentes, (2) divergentes y (3) secuenciales.

Los dos últimos casos de tipos de arcos son indistinguibles en base a pruebas de independencias, ya que son equivalentes. En ambos, los nodos X y Z son independientes dado Y. El primer caso es diferente, ya que X y Z son dos nodos padres que son marginalmente independientes. Es este primer paso el que se puede utilizar para determinar las direcciones de los dos arcos que unen estas tres variables, y a partir de éstos, es posible encontrar las direcciones de otros arcos utilizando pruebas de independencia.

El algoritmo para el aprendizaje de la estructura en poliárboles se presenta en la tabla 3.4. el cual, solamente puede ser utilizando con este tipo de redes y no siempre garantiza la obtención de todas las direcciones.

Tabla 3.4. Algoritmo para el aprendizaje en poliárboles.

1. Obtener el esqueleto de la red utilizando el algoritmo de Chow y Liu para árboles. 2. Recorrer la red hasta encontrar una tripleta de nodos que sean convergentes, donde la variable a la que apuntan los arcos se llamarán *nodo multipadre*. 3. A partir de un nodo multipadre, se determinan las direcciones de otros arcos utilizando la prueba de dependencia de tripletas, hasta donde sea posible (base causal). 4. Repetir los paso 2 a 3 hasta que ya no se puedan descubrir más direcciones. 5. Si quedan arcos sin direccionar, utilizar semántica externa para obtener su dirección.

3.5.2.3. Aprendizaje de redes multiconectadas

La estructura de las redes multiconectadas hace que el aprendizaje de este tipo de redes bayesianas sea el más difícil de los tres, de manera análoga sucede en cuanto a complejidad en el proceso de propagación de probabilidades.

Actualmente, existen dos clases de métodos para el aprendizaje de estructuras multiconectadas de redes bayesianas. Estos son:

1. Métodos basados en medidas de "Score & Search" (ajuste y búsqueda).

2. Métodos basados en pruebas de independencia.

Ambas medidas son semejantes pero el método MDL es algo mejor, ya que, suele crear estructuras más simples.

Una alternativa a los métodos anteriores es esbozar nuevamente el problema de aprendizaje como un caso de optimización, intentando de encontrar la estructura que ocasione un rendimiento deseable con el mínimo número de arcos (49).

En la tabla 3.5., se describe el algoritmo alternativo, que partiendo del árbol inicial del algoritmo de Chow y Lui se van agregando arcos hasta llegar al rendimiento deseado o al máximo número permitido. El único problema del empleo de este algoritmo es que al asumir una variable hipótesis, la orientación de los arcos será arbitraria con lo que no se puede reflejar, necesariamente, la causalidad entre los nodos.

Tabla 3.5. Algoritmo alternativo en redes multiconectadas

1. Obtener una estructura de árbol inicial mediante el algoritmo de Chow y Liu. 2. Hacer la variable hipótesis el nodo raíz y a partir de éste determinar la direccionalidad de los arcos. 3. Producir el ordenamiento de los nodos $\{X_1, X_2,\ldots,X_n\}$ a partir de la raíz y siguiendo el árbol de acuerdo a la información mutua entre variables. 4. Probar la capacidad predictiva del sistema: a. Si es satisfactorio, se finaliza el algoritmo. b. Si no, se agrega un arco y se repite el paso 4. Se seleccionará el arco de mayor información mutua basando su dirección en el ordenamiento anterior, de forma que su nodo inicial sea anterior al nodo final.

A pesar de los dos métodos proporcionados, encontrar la estructura óptima de una red bayesiana, es un proceso difícil ya que el espacio de búsqueda suele ser muy grande debido al crecimiento exponencial del número de estructuras diferentes que se pueden obtener con unas pocas variables. Por eso se suelen emplear a la vez estrategias de búsqueda heurística que encuentren una solución aceptable pero, generalmente, no óptima; siendo otra alternativa el combinar los métodos automáticos con conocimiento de expertos (46).

A continuación, se desarrollarán las dos métodos expuestos como técnicas de aprendizaje estructural de la redes multiconectadas.

3.5.2.3.1. Aprendizaje basado en "Score-Search"

Este método de aprendizaje se basa en un análisis de la situación global de la estructura de la red respecto de los datos. Como visión general se puede decir que es un proceso donde se irán generando diferentes estructuras y se evaluarán respecto a los datos utilizando medidas de ajuste. Básicamente existen dos tipos de métodos parecidos que dependen principalmente de dos aspectos principales. El primero se basa en medidas de ajuste de la estructura a los datos y el segundo en métodos de búsqueda de la mejor estructura; de ahí el nombre de medidas de "*Score and Search*".

En el método de ajuste existen principalmente dos medidas que son las más usadas. Estas son la media bayesiana y la medida basada en la longitud de descripción mínima (del inglés *Minimum Description Length, MDL*).

La medida bayesiana estima la probabilidad de la estructura dado los datos y trata de maximizarla. Esto es:

$$P(B_s|D)$$

siendo B_s la estructura de la red y *D* los datos. Como el objetivo es comparar dos estructuras, para saber cuál es mejor, habrá que escribir la fórmula anterior en términos relativos a la comparación entre dos estructuras. Si denominados *i* y *j*, a las dos estructuras se tendrá entonces:

$$P(B_{si}|D)/P\big(B_{sj},D\big) = P(B_{si},D)|P(B_{sj},D)$$

Si se consideran variables discretas y datos independientes, entonces las dos estructuras se pueden comparar en función del número de ocurrencias de los datos predichos por cada estructura.

La medida MDL (50) calcula la longitud requerida para representar la probabilidad conjunta de una estructura. En este proceso se realizan dos cálculos en la red que implican por un lado la exactitud y por otro la complejidad del modelo. La exactitud se calcula midiendo la información mutua entre los atributos y la clase. En cambio, la complejidad se estima contando el número de parámetros.

Se emplea una constante α, en [0,1], como balanceador entre exactitud y complejidad. Así, la medida de calidad quedará definida por:

$$MC = \alpha(W/Wmax) + (1-\alpha)(1 - L/Lmax)$$

donde *W* es la exactitud del modelo y *L* la complejidad, mientras que *Wmax* y *Lmax* representan la máxima exactitud y complejidad, respectivamente, del modelo. Para calcular los máximos normales se considera una limitación en cuanto al número de padres máximo permitido por nodo. El valor medio para *α,* que aporta la misma importancia entre complejidad y exactitud, es de 0,5, mientas que un valor cercano a 0

considera una mayor importancia a la complejidad, y por último, valores cercanos a 1 darán mayor importancia a la exactitud.

La exactitud se puede calcular en base al 'peso' de cada nodo, donde el peso, *xi,* se calcula en base a la información mutua con los padres, *Fxi*, a través de:

$$w(xi, Fxi) = \sum_{xi} P(xi, Fxi)\log\left[P(xi, Fxi)/P(xi)P(Fxi)x\right]$$

donde el peso (exactitud) total viene dado por la suma de los pesos de cada nodo:

$$W = \sum_{i} w(xi, Fxi)$$

La complejidad viene dada por el número de parámetros requeridos para representar el modelo, la cual se puede calcular con la ecuación:

$$L = S_i[k_i log_2 n + d(S_i - 1)F_i]$$

Donde, *n* es el número de nodos, *k* es el número de padres por nodo, S_i es el número de valores promedio por variable, F_i el número de valores promedio de los padres, y *d* el número de bits por parámetro.

Por último, a través de un método de búsqueda que encuentre la mejor estructura de red entre todas las combinaciones posibles de estructuras y, empleando la medida de calidad, vista anteriormente, se finaliza el proceso de aprendizaje de redes multiconectadas.

El método de búsqueda por lo general se basa en búsquedas heurísticas que agilizan el proceso, mejorando la eficiencia. Un ejemplo común es el empleo del algoritmo de búsqueda "*Hill Climbing"* (ascenso de colinas), el cuál a partir de una estructura simple en forma de árbol va mejorando hasta llegar a una estructura mejor.

En la tabla 3.6., se muestre el algoritmo de búsqueda de la "mejor" estructura para redes multiconectadas, que como ya se comentó anteriormente, no garantiza encontrar la estructura óptima, debido a que se puede dar una situación de máximo local.

Tabla 3.6. Algoritmo de búsqueda de estructura en redes multiconectadas.

1. Generar una estructura inicial con tipología de árbol.
2. Realizar el cálculo de la medida de calidad de la estructura actual.
3. Agregar o invertir la direccionalidad en un arco en la estructura actual.
4. Calcula nuevamente la calidad de la nueva estructura creada.
5. Si la calidad mejora se conserva el cambio, si por el contrario esta calidad empeorase se deja la estructura anterior al cambio.
6. Se repiten los paso 3 a 5 iterativamente hasta que no haya más mejoras posibles.

3.5.2.3.2. *Aprendizaje basado en pruebas de independencia*

Este método, a diferencia del método anterior, se encarga de explorar las relaciones de dependencia existentes entre pares, tripletas y otros subconjuntos de variables para seleccionar la manera en que deben de ser conectadas. El estudio de estas relaciones requerirá establecer unos criterios cuantitativos para poder calcular la dependencia entre variables, siendo este criterio principal criterio de guían en la construcción de la topología de la red.

El algoritmo de Chow y Liu, es un caso sencillo de este tipo de método, ya que se sigue un proceso en el cuál se calcula la información mutua entre pares de variables. Es a partir de este cálculo, donde se inicia el criterio para generar la red bayesiana en forma de árbol. Si el análisis de dependencias se realiza entre tripletas de variables, el método se extiende también a poliárboles.

Como ejemplo se puede mencionar el clasificador TAN, visto en el apartado, 3.4.2., pues posee un algoritmo que se engloba en este tipo de técnicas de construcción.

3.6. Redes Bayesianas Dinámicas

Las redes bayesianas representan un razonamiento probabilístico basado en el contexto de mundos estáticos, en los que cada variable aleatoria tiene un solo valor fijo. Para poder representar procesos dinámicos de modelos probabilísticos temporales, existe una extensión a las redes bayesianas conocidas con el nombre de red bayesiana dinámica, RBD (en inglés "*Dynamic Bayesian Network*")[4]. Estas redes se basan en la discretización del tiempo y en crear una réplica de cada variable aleatoria para cada punto temporal.
La representación de las redes bayesianas dinámicas se basa generalmente en las siguientes suposiciones:

- Suposición markoviana: el futuro es condicionalmente independiente del pasado dado el presente. (Esta propiedad no debe ser confundida con la condición de Markov que se aplica en la definición formal de una red bayesiana, aunque ambas estás muy relacionadas (51)).

- Proceso estacionario en el tiempo: las probabilidades condicionales en el modelo no cambian con el tiempo.

Las dos suposiciones anteriores, implican que se pueda definir una RBD en base a dos componentes. El primero, como una red base estática que se repite en cada periodo, de acuerdo a cierto intervalo de tiempo predefinido y el segundo, en base a una red de transición entre etapas consecutivas.

[4] También pueden ser llamadas con el nombre de redes de creencia dinámicas (en inglés, *dynamic belief networks*), redes temporales probabilísticas (en inglés, *probabilistic temporal networks*) y redes probabilísticas causales dinámicas (en inglés, dynamic causal probabilistic networks) (31).

3.6.1. Construcción de una RBD

Antes de comentar el proceso de construcción de una RBD, se dará una serie de explicaciones previas sobre los nodos, la estructura y las tablas de probabilidad condicionada, que servirán de ayuda al lector para entender posteriormente, todos aquellos aspectos de las RBD que se tratarán en este y en sucesivos apartados. El apartado 3.2. recoge el proceso de construcción de redes bayesianas causales, por lo que se recomienda primeramente revisar esta sección para posteriormente seguir con la construcción de una RBD.

Si se supone que el domino, que se está modelando, está compuesto por un conjunto de *n* variables aleatorias $X = \{X_1, \ldots, X_2\}$, siendo cada variable X_i es representada por un nodo en la red bayesiana, cuando se construye una RBD para modelos temporales, el estado actual de la red será representada por *t,* el estado de tiempo anterior será representado por *t-1* y el próximo estado de tiempo será *t+1*, tal que cada nodo de la RBD correspondientes serán:

- Estado actual: $\{X_1^t, X_2^t, \ldots, X_n^t\}$
- Estado anterior: $\{X_1^{t-1}, X_2^{t-1}, \ldots, X_n^{t-1}\}$
- Estado siguiente: $\{X_1^{t+1}, X_2^{t+1}, \ldots, X_n^{t+1}\}$

Cada estado de tiempo o cambio de tiempo se denominan cortes de tiempo. La relaciones entre variables en un mismo corte de tiempo viene representado por $X_i^t \rightarrow X_j^t$. Aunque se ha considerado esta representación entre variables, la estructura en el mismo corte de tiempo no suele cambiar en el tiempo, es decir, la relación entre las variables $X_1^t, X_2^t, \ldots, X_n^t$ es la misma a pesar de *t.*

Las relaciones entre variables de cortes de tiempo sucesivos están representadas por arcos temporales que pueden ser relaciones entre la misma variable en el tiempo, $X_i^t \rightarrow X_i^{t+1}$, o entre diferentes variables en el tiempo, $X_i^t \rightarrow X_j^{t+1}$.

En la mayoría de los casos, el valor de una variable en un tiempo *t* afecta a su valor siguiente en *t+1* de manera circular, $X_i^t \rightarrow X_i^{t+1}$, de tal modo que casi siempre este tipo de arcos estarán presentes en la estructura de la red. Aunque en general, el valor de cualquier nodo puede afectar al valor de cualquier otro nodo en el instante de tiempo siguiente.

La figura 3.13 muestra un ejemplo de RBD, en este caso se muestra una estructura base que se repite sucesivamente *t* etapas temporales, así como las relaciones de dependencia entre etapas. Los cortes de tiempo, son representados por los cuadrados con líneas discontinuas.

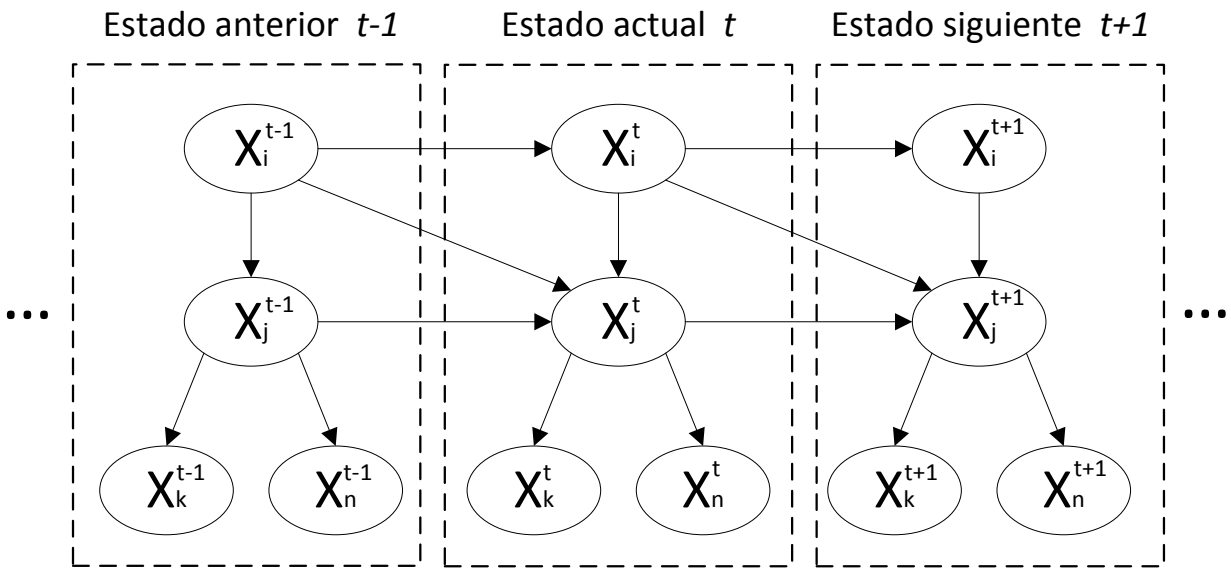

Figura 3.13. Ejemplo general de la estructura de una RBD.

Las relaciones entre las variables, tanto entre cortes de tiempo como entre variables de un mismo corte de tiempo, se cuantifican mediante la distribución de probabilidad condicional asociada a cada nodo.
En general, para un nodo X_1^t, con padre en un mismo corte de tiempo $Y_1^t, \dots, Y_m^t$, y entre padres de diferente corte X_1^{t-1}, y $Z_1^{t-1}, \dots, Z_k^{t-1}$, la tabla de probabilidad condicionada es:

$$P(X_i^t | Y_1^t, \dots, Y_m^t, X_1^t, Z_1^{t-1}, \dots, Z_k^{t-1})$$

Dada la restricción habitual de que las redes por cada corte de tiempo son exactamente iguales y que los cambios en el tiempo también son los mismos (es decir, tanto la estructura de la red como las tablas de probabilidad condicionada son inalterables), una RBD puede ser construida de forma muy compacta.

La construcción de una RBD constará de los siguientes pasos:

- Definir los nombres de los nodos.
- Crear los enlaces entre nodos del mismo corte de tiempo.
- Crear los arcos temporales entre nodos de distinto corte de tiempo.
- Especificar la distribución *a priori* sobre las variables de estado, $P(X_i^t)$.
- Especificar las tablas de probabilidad condicionada para los nodos en rebanadas de tiempo *t = 0,* $P(X_i^t | X_i^t)$ (aquellos nodos que no tienen un padre del tiempo anterior).
- Especificar las tablas de probabilidad condicionada para los nodos en rebanadas de tiempo *t+1,* $P(X_i^{t+1} | X_i^t)$.

Debido a que las probabilidades $P(X_i^t)$, $P(X_i^t | X_i^t)$ y $P(X_i^{t+1} | X_i^t)$, se suponen que son estacionarios (el mismo para todo t) es sencillamente más conveniente especificarlos para el primer corte. Ya que a partir de esta especificación, la RBD completa (semi-infinita) puede construirse, cuando sea necesario, copiando el primer corte. (32).

3.6.2. Inferencia en RBD

El proceso de inferencia se describió en el apartado 3.3., donde se ven las distintas técnicas de propagación en los distintos tipos de redes que se pueden encontrar. En principio la inferencia en RBDs es la misma que para redes bayesianas, por lo que se aplican los mismo métodos. Sin embargo, al tratar con RBD la complejidad de la estructura aumenta, por lo que los métodos más comunes son aquellos basados en simulaciones estocásticas, como el método de Monte Carlos también denominado como filtro de partículas.

El problema general de la inferencia en RBDs es calcular $P(X_{i:t0}|E_{t1:t2})$, donde $X_{i:t0}$ representa la variable oculta i-enésimo en el tiempo y $E_{t1:t2}$ representa a todas las evidencias entre los tiempo *t1* y *t2.* La observación en el instante *t* es $E_t = e_t$ para algún conjunto de valores e_t.
Existen varios casos especiales de interés a la hora de formular las tareas de inferencia básica que pueden ser considerados (6), (33). Estas tareas son:

- ***Filtrado***: esta es la tarea de calcular el estado de creencia (la distribución a posteriori del estado actual, dada toda la evidencia hasta el momento). Esto consiste en calcular $P(X_t|e_{1:t})$, suponiendo que la evidencia a parece en un flujo continua desde *t* = 1.

- ***Predicción***: esta tarea consiste en calcular la distribución a posterior del estado futuro, dada toda la evidencia hasta el momento. Esto consiste en calcular $P(X_{t+k}|e_{1:t})$ para algún *k* > 0.

- ***Suavizado:*** esta tarea consiste en calcular la distribución a posteriori del estado pasado, dada toda la evidencia hasta el momento presente. Esto consiste en calcular $P(X_k|e_{1:t})$ para algún *k* tal que *0 ≤ k < t*.

- ***Explicación más creíble:*** dada una secuencia de observaciones, se puede querer encontrar la secuencia de estados más creíbles que hayan generado esas observaciones. Esto consiste en calcular $argmax_{x_{1:t}} P(X_{1:t}|e_{1:t})$

La figura 3.14. (ver página siguiente) representa visualmente los principales tipos de inferencia presentados anteriormente. La región sombreada es el intervalo para el que se dispone de datos. La flecha, representa el paso de tiempo en el que se desea realizar la inferencia. *t* representa el instante de tiempo y *T* es la longitud de la secuencia.

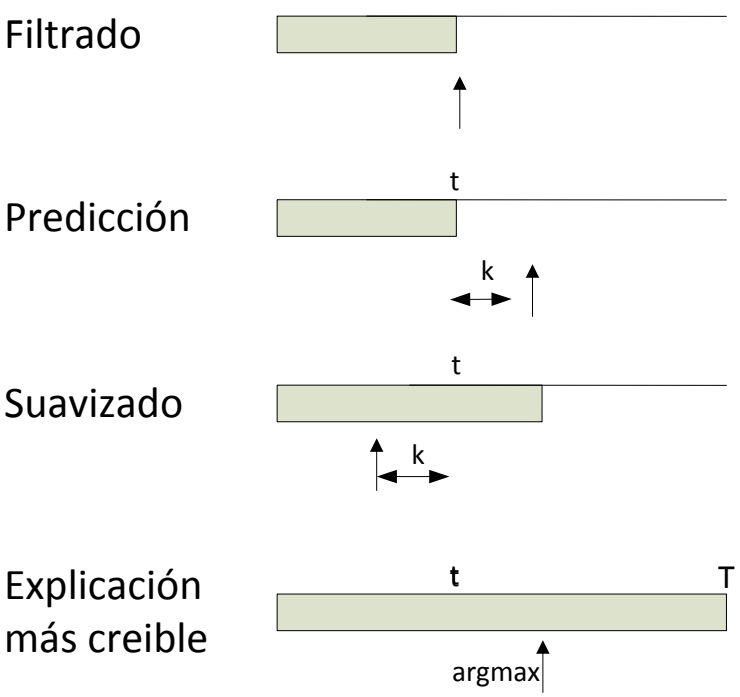

Figura 3.14. Representación de los principales tipos de inferencia en RBDs (6).

Teniendo en cuenta la evidencia sobre un conjunto de nodos, $E_{1:t}$, en el primer corte de tiempo, hasta un corte actual *t,* el proceso de actualización de las evidencia en la RBD completa se puede realizar a través de algoritmos de inferencia estándares de las redes bayesianas. Esto significa que dada una secuencia de observaciones, uno puede construir la representación de la red bayesiana completa duplicando cortes hasta que la red sea lo suficientemente grande como para adecuarse a las observaciones, para posteriormente una vez completa la RBD usar cualquiera de los algoritmos de inferencia(claro está acorde con la estructura de la red) . Esta técnica se conoce como desenrollado.
Sin embargo, cuando se tiene una secuencia elevada de observaciones $e_{1:t}$, la técnica de desenrollado no es eficiente, ya que se requerirá un espacio *O(t)*, y de este modo crecería sin límites conforme se añadieran más observaciones. Más aún, si cada vez que se añade una observación simplemente se procede a ejecutar de nuevo el algoritmo de inferencia. El tiempo de inferencia por actualización también aumentará según *O(t)* (32).

Para hacer frente a estos casos en los que las RBDs se hacen muy grandes rápidamente (sobre todo cuando la duración del segmento de tiempo es corto), se incorpora un proceso de razonamiento consistente en una "ventana" deslizante, de tamaño fijo que se mueve hacia adelante con el tiempo. Cuando se desplaza la "ventana" hacia delante, un corte de tiempo se sale y uno nuevo es añadido.
Con este uso de "ventaja" fija, cada vez que se avanza en el tiempo las observaciones anteriormente recibidas no estarán directamente disponibles. En su lugar, se tendrá la creencia actual de los nodos, que pasará posteriormente a ser las siguientes probabilidades *a priori.*

La figura 3.15., muestra el proceso anteriormente explicado en una RBD que consta de una ventana con dos cortes de tiempo. Esta estructura puede ser considerada como un RBD genérica que consiste en el nodo estado *X,* su correspondiente nodo de observación *O*, donde la evidencia es añadida y un nodo acción *A*, que afectará al nodo de estado en el siguiente tiempo.

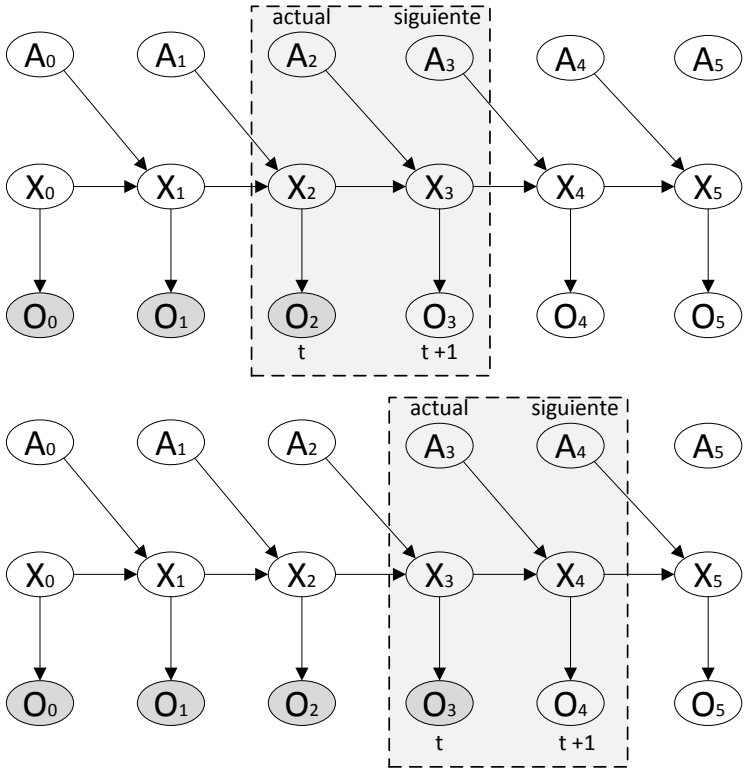

Figura 3.15. Ejemplo de ventana deslizante de dos cortes de tiempo en una RBD (El sombreado indica la evidencia de los nodos).

El algoritmo que realiza el proceso de actualización en una RBD, a través del proceso de la ventana "deslizante", se muestra en la tabla 3.7. Es de señalar que los pasos de actualización de este algoritmo son exactamente similares a los de la técnica utilizada en la teoría de control básico, llamado filtro de Kalman.

Tabla 3.7. Proceso de actualización de una RBD.

1. **Desplazamiento**: mover la ventana
2. **Predicción:**
   a. Se calcula la probabilidad de $P\left(X_{t-1}\middle|E_{\{1,t-1\}}\right)$, que es la probabilidad a posteriori sobre X_{t-1}.
   b. Se estima la predicción de $P\left(X_t\middle|E_{\{1,t-1\}}\right)$
3. **Desenrrollamiento:**
   a. Eliminar el corte de tiempo para *t-1*.
   b. Utilizar las predicciones para el segmento de *t*, como la nueva configuración previa por parte de $P(X)$ para $P\left(X_t\middle|E_{\{1,t-1\}}\right)$
4. **Estimación:**
   a. Añadir la nueva observación E_t.
   b. Calcular la probabilidad de $P\left(X_t\middle|E_{\{1,t\}}\right)$, que es la probabilidad a posteriori del estado actual.
   c. Añadir el corte de tiempo para *t+1*.

A continuación, se muestra un modelo de razonamiento probabilístico en el tiempo, que es una singularización de las RBDs, pues está incluye a los modelos ocultos de Markov (en inglés "*Hidden Markov Models"*, HMMs) como caso particular.

3.6.3. Modelos ocultos de Markov

Los modelos ocultos de Markov es el tipo más simple de DBN con el que uno se pude encontrar. Y se puede definir como un autómata finito estocástico, donde cada estado del proceso está definido por una única variable aleatoria discreta, donde los valores posibles de la variable son los estados posibles del mundo.

Un HMM queda definido principalmente por los siguientes elementos: un conjunto *S* de estado posibles, el modelo de transición $P(X_t|X_{t-1})$ se convierte en una matriz *T* de dimensión *SxS*, donde $T_{ij} = P(X_t = j|X_{t-1} = j)$ (siendo la transición desde el estado *i* hasta el estado *j*) y por último la matriz diagonal O_t,(construida cada paso de tiempo *t*) cuyas entradas diagonales están dada por los valores $P(e_t|X_t = i)$ siendo el resto de entradas 0 y e_t un valor conocido de la variable evidencia E_t.

La estructura básica o topología de un HMM queda definida por el número de estados ocultos, el número de símbolos y las transiciones de estados y emisiones de símbolos no permitidas (que como se mención anteriormente la probabilidad de esta transición se asumen que son cero).

En la figura 3.16., se puede ver una representación gráfica de una red bayesiana correspondiente a un HMM de primer orden (el estado actual depende sólo del estado previo y no de estados iniciales), representada median un grafo acíclico dirigido, donde los nodos representan las variables aleatorias(los estado ocultos sin estar sombreados y las observaciones sombreadas) y donde la ausencia de una flecha entre dos variables nos indica su independencia condicional.

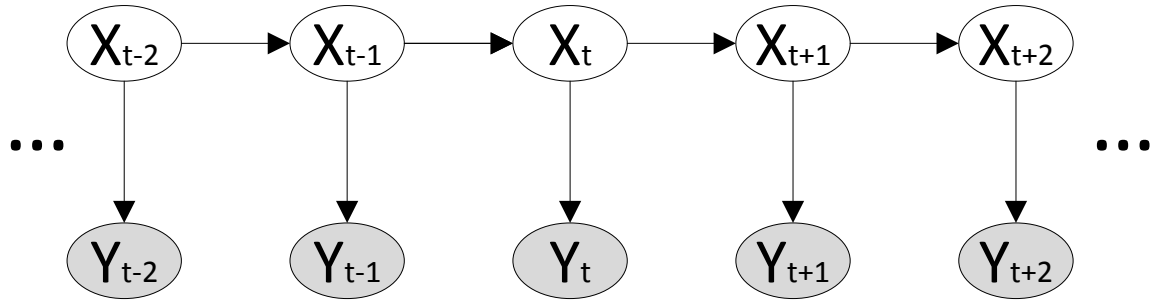

Figura 3.16. Ejemplo de un HMM de primer orden en forma de red bayesiana.

Un HMM puede ser representado como una RBD con una sola variable de estado y una sola variable de evidencia.

La diferencia entre una RBD y un HMM es que, «*en la descomposición del estado de un sistema complejo en sus variables constituyentes, la RBD es capaz de aprovecharse de la poca densidad del modelo probabilista temporal*» (32). Esto quieres decir que, si el

HMM tiene una longitud de *n* y la cardinalidad del espacio de estados (finito) tiene cardinalidad *m* entonces existen m^n posibles recorridos de la cadena oculta, lo que para valores no muy grandes de *n* y *m*, el tratamiento de estas redes se vuelve difícil de manejar. Por lo que para problemas grandes el uso de HMM será inadecuado debido principalmente a la enorme cantidad de espacio que se necesita, a la enorme matriz de transición (que se crea rápidamente debido al crecimiento exponencial) que ocasiona que el proceso de inferencia en un HMM sea mucho más costosa que en una RBD y por último, a problemas del proceso de aprendizaje debido al número de parámetros.

3.6.4. Aprendizaje en RBD

En esta sección se recogerá brevemente el aprendizaje de una RBD en base a dos componentes, la estructura base y la red de transición. En el apartado 3.5., se recogió el aprendizaje en las redes bayesianas en la exponen las técnicas que en esta sección se mencionan.

El aprendizaje de una RBD puede dividirse en el aprendizaje por separado de dos partes:

1. Aprender la estructura base (estática).
2. Aprender la estructura de transición.

Para el aprendizaje de la estructura base, se consideran los datos de todas las variables en cada tiempo, de forma que sea posible obtener las dependencias entre éstas sin considerar las relaciones temporales. Este proceso es equivalente al aprendizaje estructural y al aprendizaje paramétrico de una red bayesiana (ver apartado 3.5.), y por consiguiente se pueden aplicar cualquiera de los métodos que se mencionan.
Una vez obtenida la estructura base, el siguiente paso es la obtención de la red de transición. Este aprendizaje se puede realizan usando el enfoque basado en medidas de "score and search" o el enfoque de medidas locales (ver apartado 3.5.).

Para el enfoque basado en "score and search", se parte de una estructura inicial con dos copias de la red base, y se busca agregar las relaciones entre variables en el tiempo *t* y *t+1* que optimicen la medida de evaluación. En este proceso se consideran los datos de cada variable en un tiempo y en el siguiente. Para el enfoque de medidas locales, se aplican éstas a las variables entre etapas para de esta forma determinar los arcos a incluir en la red de transición.

La figura 3.17. (ver página siguiente) muestra el esquema general de aprendizaje de una RBD para un ejemplo sencillo donde primero se obtiene la estructura base (grafo izquierdo) y después las relaciones entre etapas (grafo derecho).

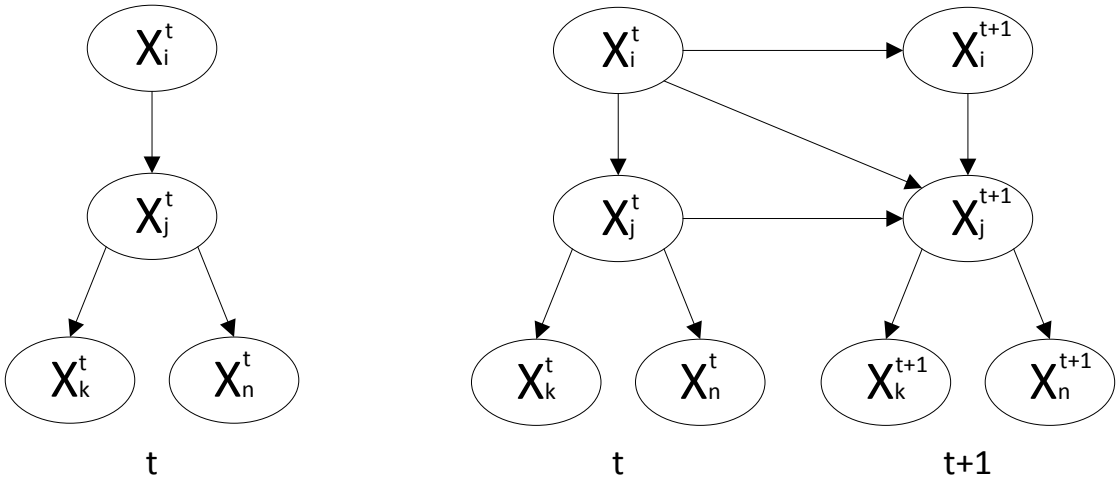

Figura 3.17. Ejemplo del aprendizaje de una RBD.

3.7. Ventajas e inconvenientes de las redes bayesianas

Hasta el momento se ha visto la base teórica de las redes bayesianas sin adentrarse en los beneficios o perjuicios que aportan éstas al ser una combinación de técnicas estadísticas y modelos gráficos. Es por eso, que en este apartado se dedicará a recoger todas las ventajas e inconvenientes de las redes bayesianas.

El hecho de que las redes bayesianas tengan una clara interpretación, hace que sea posible combinar datos estadístico y datos estimados de forma subjetiva, siendo posible justificarlas sobre bases teóricas sólidas. A pesar de ello, la desventaja que conlleva es que el método de razonamiento llevado a cabo se basa en una aproximación normativa, basado en una teoría matemática y, en consecuencia, la comprensión del proceso de inferencia es más difícil que en los métodos que tratan de imitar el razonamiento humano (38).

La propiedad de almacenar información sobre las dependencias e independencias entre variables permite manejar situaciones con incertidumbre, que unido a la representación gráfica de la red facilita la interpretación y la obtención de conclusiones por parte del observador.

Como se ha visto en el proceso de construcción, en las redes bayesianas, las relaciones de causalidad, unido con la lógica probabilística permite combinar el conocimiento extraído de expertos con datos. Sin embargo, esto conlleva que en el proceso de construcción manual de redes bayesianas, la parte más compleja, sea la asignación de valores a los parámetros, que en muchas ocasiones, no estarán disponibles y por consiguiente tendrán que ser objeto de estudio.

El empleo del razonamiento causal es una de las características esencial de las redes bayesianas que aportan un conjunto de ventajas importantes frente a otro sistemas. La primera de ellas es que el diagnóstico que realizan se basa en un modelo explícito que tiene en cuenta la influencia entre causas y efectos. Además una red bayesiana no

solamente es capaz de explicar cómo se ha realizado la inferencia, sino también el modelo en sí, independientemente de la evidencia observada.
Otra ventaja del empleo del razonamiento causal es que la red admite varios tipos de inferencias:

- Razonamiento abductivo: es aquel que busca cuál es la causa o la combinación de causas que mejor explica (aquella que mayor probabilidad tenga) dado un hallazgo.

- Razonamiento predictivo: es aquel que se produce cuando se da una anomalía, pudiendo realizar una estimación de la probabilidad con el objetivo de dar un pronóstico o de encontrar el resultado más probable en un futuro.

- Razonamiento intercausal: que se da cuando hay dos causas de un mismo efecto. La ausencia de una de ellas nos hará sospechar de la presencia de la otra, y viceversa: si una de ella es explicación suficiente del hallazgo, entonces éste ya no es motivo para sospechar la presencia de la otra, aunque tampoco la descarta (esta propiedad es conoce con el nombre en inglés de *explaining away*).

A pesar del conjunto de ventajas que aporta el trabajar con redes bayesianas existen sin embargo, inconvenientes en el empleo de esta metodología.

La complejidad de los algoritmos es uno de los mayores problemas ya que el cálculo de probabilidad en redes que presenten bucles es un problema NP (todavía no existe una demostración completa de la existencia de un algoritmo polinómico en función del tamaño de la red). Esto significa que el tiempo necesario para estimar la probabilidad en una red bayesiana crece de forma exponencial con el número de nodos y enlaces.

Como ya se comentó en el apartado 3.2.1. Construcción manual, el proceso de obtención del conocimiento requiere una gran cantidad de tiempo que hace que muchas veces sea un proceso complicado tanto por la falta de expertos como por la inexistencia de bases de datos con las variables adecuadas que permitan obtener el gran número de probabilidades *a priori* y condicionales. Se ha considerado el problema anterior como inconveniente aunque realmente es una dificultad intrínseca de los problemas que se abordan.

La ventaja de la causalidad de las redes bayesianas, implica como inconveniente que en dominios donde las relaciones causales no estén bien definidas resulte difícil construir un sistema experto. Esto conlleva a una limitación en el rango de aplicaciones que se pueden construir con esta técnica. Las redes bayesianas están indicadas en problemas de diagnóstico, y por consiguiente en las tareas relacionadas, como la interpretación, la predicción y monitorización. Sin embargo, en problemas de diseño, planificación, instrucción, reparación y control, las redes bayesianas encuentran serias dificultades, pues en ellos desempeñan un papel muy importante el razonamiento temporal y la toma de decisiones.

Capítulo IV. Objetivos

En este apartado se realiza la adquisición de los objetivos que se quieres conseguir del proyecto. En el capítulo anterior se ha visto las definiciones y características de las redes bayesianas, y en especial de las RBD sobre los que se trabajará posteriormente a los largo del desarrollo del proyecto. El objetivo principal consistirá en reproducir el proceso de inferencia temporal de las RBD, para así poder crear un entorno de usuario que se pueda aplicar a distintos problemas de fusión de información. Será pues a través de esta interfaz a partir del cual, se ofrecerá una herramienta que permita realizar razonamientos en el tiempo con flujos de información sensorial (o de otro tipo) en ámbitos distintos.

4.1. Especificación de casos de uso

Un caso de uso es la descripción de un uso típico del sistema o aplicación, que proporciona un resultado valioso para el usuario cuyo propósito consiste en describir las interacciones entre la aplicación y los agentes externos, con el fin de obtener los requisitos de la aplicación. Así pues gracias a los casos de uso se podrá comprender y criticar un escenario de cómo las personas interactúan con el sistema software.

Cuando en la definición anterior se hace uso de escenario, éste se entiende como conjuntos de descripciones de ejemplos de sesiones de interacción entre un actor y el sistema. Por lo tanto, cada escenario es algo que puede ocurrir en el sistema con el uso de éste. De modo que, un caso de uso es un conjunto de escenarios que pueden ser agrupados para la consecución de un fin común. Normalmente uno de estos escenarios representa el caso más típico del uso de una parte determinada del sistema y los demás escenarios, correspondientes al mismo caso de uso, representan usos raros o posibilidades de realizar la misma operación utilizando para ello caminos alternativos.

En los diagrama de casos de usos, además de aparecer representados los servicios ofrecidos por la aplicación, también aparecerán representados los actores. Un actor, representa el rol que adopta una entidad externa que interacciona directamente con el sistema.

Para representar de forma gráfica los casos de uso, se utiliza el lenguaje de modelado UML.

En el siguiente diagrama de casos de uso (mostrado en la página siguiente), se recoge la funcionalidad a la que puede acceder el usuario de la aplicación. Es de señalar, que algunos de los casos de uso recogidos a continuación, se descomponen realmente en varios, pero que se ha optado por esta representación ya que, debido a una correcta jerarquización de la especificación se facilita la comprensión global del modelo de casos de uso. Posteriormente, en sucesivos diagramas se recogerán los casos de uso en los que se descomponen estas jerarquías.

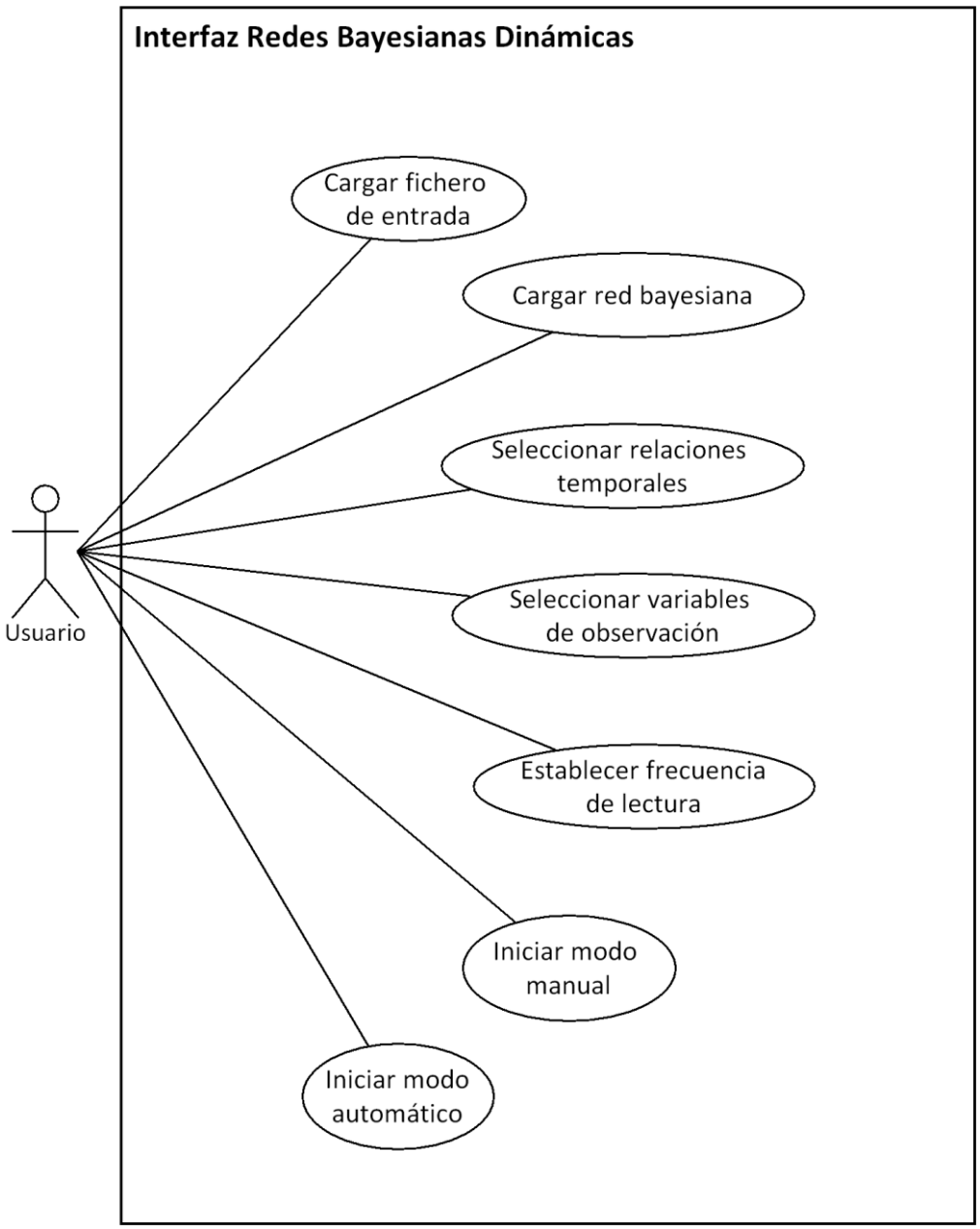

Figura 4.1. Diagrama de casos de uso del sistema.

Caso de uso Iniciar modo manual

El caso de uso iniciar modo manual hace referencia a la funcionalidad que el sistema ofrece a un usuario para interactuar a través de un uso manual en el cuál el usuario tiene que ir seleccionando las evidencias observadas en la red bayesiana dinámica la cual quiere estudiar.

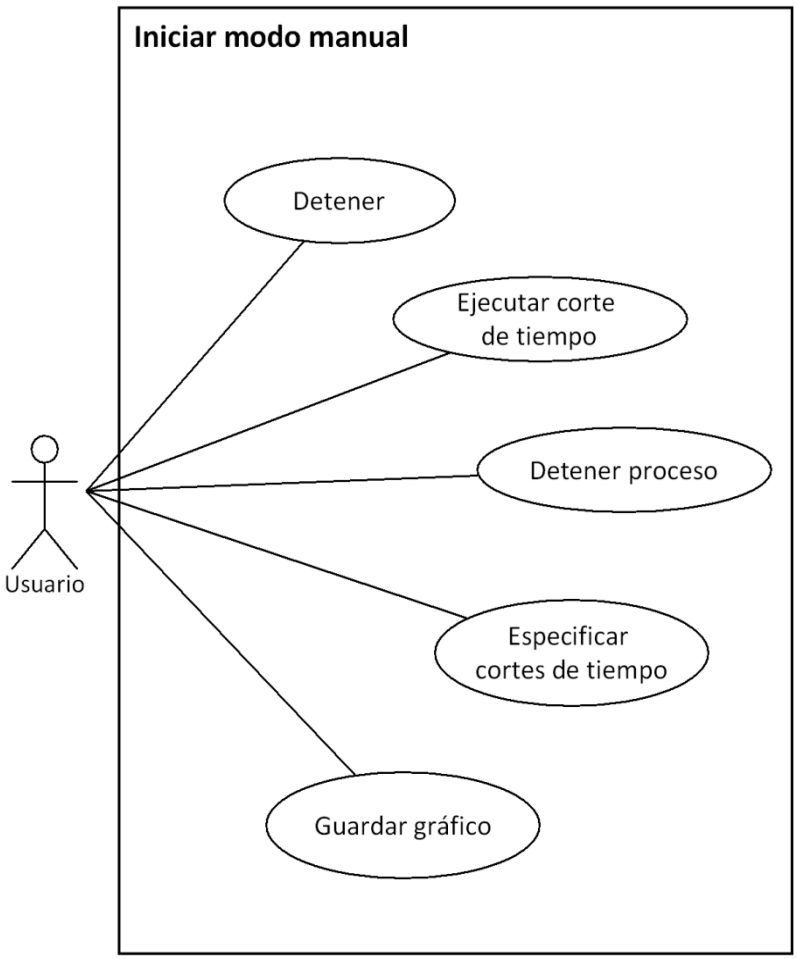

Figura 4.2. Caso de uso Iniciar modo manual.

Caso de uso Iniciar modo automático

El caso de uso Iniciar modo automático hace referencia a la funcionalidad que el sistema ofrece a un usuario para iniciar un modo sistemático en el cuál el sistema realiza las operaciones necesarias en la red bayesiana conforme a una serie de evidencias gestionadas en un fichero de entrada.

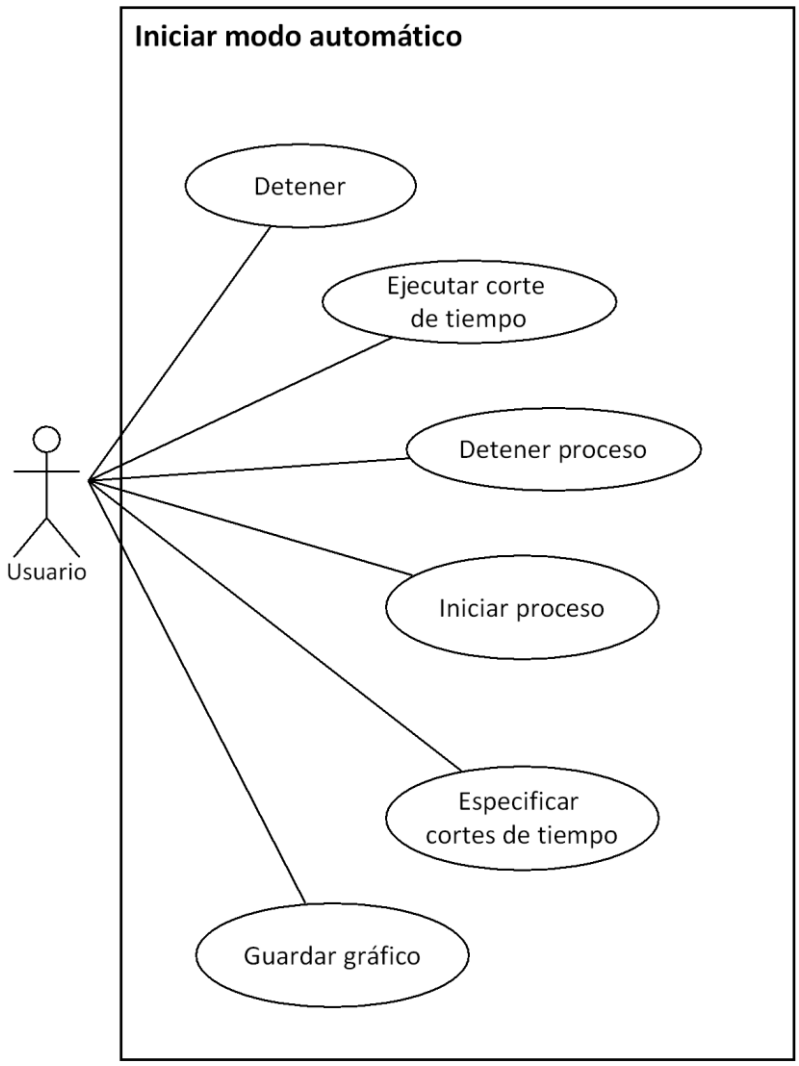

Figura 4.3. Caso de uso Iniciar modo automático.

4.1.1. Descripción textual de los casos de uso

A continuación, se realizará una descripción textual de cada uno de los casos de uso vistos en los diagramas anteriores.

Tabla 4.1. Caso de uso CU-01.

IDENTIFICADOR	CU-01
NOMBRE	Cargar fichero de entrada
ACTORES	Usuario
OBJETIVOS	Permitir al usuario introducir en la interfaz un fichero de entrada que contenga los datos sobre las evidencias de las variables de la red bayesiana
PRECONDICIONES	• La existencia de un fichero de entrada con un formato específico para la simulación automática
POSTCONDICIONES	• Se muestra la frecuencia de lectura del fichero
ESCENARIO BÁSICO	1. El usuario selecciona la opción de examinar del fichero de entrada. 2. El usuario localiza el fichero de entrada y lo carga 3. El sistema muestra la ruta del fichero cargado

Tabla 4.2. Caso de uso CU-02.

IDENTIFICADOR	CU-02
NOMBRE	Cargar red bayesiana
ACTORES	Usuario
OBJETIVOS	Permitir al usuario introducir en la interfaz una red bayesiana que permita realizar su exploración a través de las necesidades del usuario
PRECONDICIONES	• La existencia de una red bayesiana creada previamente a través de la aplicación Netica
POSTCONDICIONES	• Se muestra las relaciones existentes entre los nodos de la red bayesiana • Se muestran las variables existentes en la red bayesiana • El sistema habilita el inicio de la aplicación y muestra la ruta de la red bayesiana cargada
ESCENARIO BÁSICO	1. El usuario selecciona la opción de examinar del fichero de entrada para la red bayesiana. 2. El usuario localiza la red bayesiana y la carga

Tabla 4.3. Caso de uso CU-03.

IDENTIFICADOR	CU-03
NOMBRE	Seleccionar relaciones temporales
ACTORES	Usuario
OBJETIVOS	Permitir al usuario seleccionar las relaciones temporales existentes en la red bayesiana dinámica
PRECONDICIONES	• Haber realizado previamente la carga de una red bayesiana en la interfaz
POSTCONDICIONES	• Se muestran seleccionadas las relaciones temporales
ESCENARIO BÁSICO	1. El usuario selecciona las relaciones temporales que presenten una relación temporal

Tabla 4.4. Caso de uso CU-04.

IDENTIFICADOR	CU-04
NOMBRE	Seleccionar variables de observación
ACTORES	Usuario
OBJETIVOS	Permitir al usuario seleccionar las variables de observación para su posterior representación en gráfica
PRECONDICIONES	• Haber realizado previamente la carga de una red bayesiana en la interfaz
POSTCONDICIONES	• Se muestran las variables observadas
ESCENARIO BÁSICO	1. El usuario selecciona las variables de observación

Tabla 4.5. Caso de uso CU-05.

IDENTIFICADOR	CU-05
NOMBRE	Establecer frecuencia de lectura
ACTORES	Usuario
OBJETIVOS	Permitir al usuario establecer la frecuencia con la que el fichero de entrada de datos será leída por la interfaz
PRECONDICIONES	• Haber cargado previamente un fichero de entrada de datos en la red bayesiana
POSTCONDICIONES	• Ninguna
ESCENARIO BÁSICO	1. El usuario introduce una cantidad numérica para establecer la frecuencia de lectura del fichero de entrada 2. Confirmar la cantidad introducida 3. El sistema cambia y muestra la frecuencia de lectura
ESCENARIO ALTERNATIVO	1-2a Frecuencia de lectura inválida. 1. Aceptar notificación 2. Volver al paso 1

Tabla 4.6. Caso de uso CU-06.

IDENTIFICADOR	CU-06
NOMBRE	Iniciar modo manual
ACTORES	Usuario
OBJETIVOS	Iniciar la ejecución en modo manual de la interfaz para las redes bayesianas dinámicas
PRECONDICIONES	• Haber cargado previamente en el sistema una red bayesiana • Haber seleccionado al menos una relación temporal válida
POSTCONDICIONES	• Se representa visualmente al usuario la red bayesiana en la aplicación Netica para el uso manual de la misma
ESCENARIO BÁSICO	1. El usuario cargar una red bayesiana dinámica 2. El usuario selecciona la relaciones temporales 3. Confirmar relaciones temporales 4. El usuario inicia el modo manual 5. El sistema habilita las opciones del uso manual

ESCENARIO ALTERNATIVO	2-3a Relación temporal inválida. 1. Aceptar notificación 2. Deseleccionar la relación temporal 3. Marcar una nueva relación temporal 4. Volver al paso 3
	3-4a Selección de variables de observación. 1. Seleccionar variables de observación 2. Volver al paso 4

Tabla 4.7. Caso de uso CU-07.

IDENTIFICADOR	CU-07
NOMBRE	Iniciar modo automático
ACTORES	Usuario
OBJETIVOS	Iniciar la ejecución en modo manual de la interfaz para las redes bayesianas dinámicas
PRECONDICIONES	• Haber cargado previamente en el sistema un fichero de entrada de datos • Haber cargado previamente en el sistema una red bayesiana • Haber seleccionado al menos una relación temporal válida
POSTCONDICIONES	• Se representa visualmente al usuario la red bayesiana en la aplicación Netica • Se van actualizan las variables dependiendo de las evidencias del fichero de entrada con una frecuencia constante
ESCENARIO BÁSICO	1. El usuario carga un fichero de entrada de datos 2. El usuario cargar una red bayesiana dinámica 3. El usuario selecciona la relaciones temporales 4. Confirmar relaciones temporales 5. El usuario inicia el modo automático
ESCENARIO ALTERNATIVO	3-4a Relación temporal inválida. 1. Aceptar notificación 2. Deseleccionar la relación temporal 3. Marcar una nueva relación temporal 4. Volver al paso 4
	3-4a Modificar frecuencia de lectura 1. El usuario introduce un valor de frecuencia de lectura 2. Confirmar el valor introducido 3. Volver al paso 4
	3-4a Selección de variables de observación. 3. Seleccionar variables de observación 4. Volver al paso 4

Tabla 4.8. Caso de uso CU-08.

IDENTIFICADOR	CU-08
NOMBRE	Detener
ACTORES	Usuario
OBJETIVOS	Detener el funcionamiento de la interfaz
PRECONDICIONES	• Haber iniciado la aplicación en modo manual o en modo automático • Estar la aplicación actualmente inactiva sin realizar ningún proceso
POSTCONDICIONES	• Se detiene el funcionamiento de la aplicación • Se cierra la aplicación Netica dejando de estar visible la red bayesiana • Se vuelve al estado inicial antes de iniciar el modo manual o el modo automático
ESCENARIO BÁSICO	1. El usuario presiona sobre la opción en la interfaz de detener el sistema 2. El sistema vuelve al estado inicial hasta de su iniciación

Tabla 4.9. Caso de uso CU-09.

IDENTIFICADOR	CU-09
NOMBRE	Ejecutar corte de tiempo
ACTORES	Usuario
OBJETIVOS	Iniciar la ejecución en modo manual de la interfaz para las redes bayesianas dinámicas
PRECONDICIONES	• Haber ejecutado la iniciación en modo manual o automático del sistema • Estar la interfaz en un estado inactivo sin realizar ninguna otra tarea
POSTCONDICIONES	• Se actualizan las probabilidades a priori y condicionales de la red bayesiana acordes a un salto de tiempo • Se actualiza el tiempo general • Se devuelve al usuario el control
ESCENARIO BÁSICO	1. El usuario realiza las consideraciones apropiadas en la red bayesiana sobre las evidencias de las variables 2. El realiza la ejecución de un corte de tiempo

Tabla 4.10. Caso de uso CU-10.

IDENTIFICADOR	CU-10
NOMBRE	Detener proceso
ACTORES	Usuario
OBJETIVOS	Detener la ejecución de alguna actividad primaria del sistema que requiera el bloqueo de la interfaz y así poder regresar el control de programa al usuario
PRECONDICIONES	• Estar realizando una actividad bloqueante
POSTCONDICIONES	• Se devuelve el estado control al usuario • Se detienen las actividades que se estaban actualmente realizando • Se habilitan nuevas acciones de uso del sistema
ESCENARIO BÁSICO	1. El usuario detiene el proceso actual presionando sobre la opción habilitada para ello 2. El sistema habilita las opciones correspondientes tras la detención

Tabla 4.11. Caso de uso CU-11.

IDENTIFICADOR	CU-11
NOMBRE	Especificar cortes de tiempo
ACTORES	Usuario
OBJETIVOS	Poder realizar un número definido de inferencias en la red bayesiana a través de valor especificado por el usuario
PRECONDICIONES	• Haber ejecutado la iniciación en modo manual o automático del sistema • Estar la interfaz en un estado inactivo sin realizar ninguna otra tarea
POSTCONDICIONES	• Se actualizan las probabilidades a priori y condicionales de la red bayesiana acordes con la cantidad indicada por el usuario de número de cortes de tiempo • Se actualiza el tiempo general • Se devuelve al usuario el control
ESCENARIO BÁSICO	1. El usuario introduce una valor en la casilla especificada para realizar un número concreto de cortes de tiempo 2. Se verifica la cantidad introducida 3. Se inicia la ejecución de los coretes de tiempo 4. Se devuelve el control al usuario
ESCENARIO ALTERNATIVO	1-2a Número invalido de cortes de tiempo 1. Aceptar la notificación 2. Volver al paso 1

Tabla 4.12. Caso de uso CU-12.

IDENTIFICADOR	CU-12
NOMBRE	Guardar gráfico
ACTORES	Usuario
OBJETIVOS	Permitir al usuario de salvaguardar la representación gráfica en Gnuplot
PRECONDICIONES	• Haber iniciado la aplicación en modo manual o automático • Haber seleccionado antes del inicio de la aplicación las variables de observación • Estar el sistema en reposo sin la realización de ninguna actividad
POSTCONDICIONES	• Se guarda un fichero de datos con los valores de las variables de observación preseleccionadas a través del tiempo • Se guarda un fichero de representación en Gnuplot de las variables
ESCENARIO BÁSICO	1. El usuario presiona sobre la opción de guardar gráfico 2. El usuario introduce el nombre del fichero 3. El sistema realiza el salvaguardado de los dos ficheros

Tabla 4.13. Caso de uso CU-13.

IDENTIFICADOR	CU-13
NOMBRE	Iniciar proceso
ACTORES	Usuario
OBJETIVOS	Detener la ejecución de alguna actividad primaria del sistema que requiera el bloqueo de la interfaz y así poder regresar el control de programa al usuario
PRECONDICIONES	• Estar el sistema en el modo automático de ejecución • Haber detenido previamente el proceso automático
POSTCONDICIONES	• Se reinicia el modo de ejecución automático • Se deshabiliten ciertas opciones de uso
ESCENARIO BÁSICO	1. El usuario presiona sobre el botón habilitado para reiniciar el modo de ejecución automático 2. El sistema retoma el modo de ejecución

4.2. Requisitos de usuario

A continuación, se realizará el proceso de compresión y formalización de las necesidades que debe satisfacer el sistema a través de la captura de los requisitos de usuario, definiendo qué debe de hacer la interfaz a desarrollar.

Los requisitos se clasificarán en 2 tipos:

- Requisitos funcionales
- Requisitos de restricción

Ambos tipos de requisitos se representarán mediante una tabla cuyos contenidos serán:

- Identificador: Estará formado por RF-XXX o RR-XXX donde XXX representa un número entero para identificar los requisitos unívocamente y las iniciales RF y RR hacen referencia, respectivamente, a un requisito funcional y a un requisito de restricción.

- Prioridad: se introducirá una medida de prioridad con el fin de que el desarrollador pueda decidir el orden de producción.

- Fuente: Persona, documento o entidad a través del que se ha obtenido el requisito de usuario.

- Autor: Persona o entidad encargado de la descripción del requisito de usuario.

- Fecha: indica el momento dentro del tiempo, de la descripción del requisito.

- Necesidad: Los requisitos podrán ser: esenciales (para los cuales no existirá negociación), recomendables y optativos (cuyo cumplimiento añade bonificaciones extras a la retribución del proyecto).

- Verificabilidad: La capacidad del requisito para comprobarse que el sistema final lo cumple o incorpora. Para ello debe ser posible:
  - Comprobar que el requisito ha sido incorporado en el diseño.
  - Probar que el software implementará el requisito.
  - Testear el requisito cuando está implementado por el software.

- Estabilidad: Los requisitos se clasificarán en estables o inestables dependiendo de la previsión de que cambien o no, a lo largo del ciclo de vida del proyecto.

- Estado: indica la situación en la que se encuentra dicho requisito, siendo dos los posibles valores: implementado o no implementado.

- Descripción: Detalla brevemente el requisito de usuario.

- Dependencias: En este apartado se reflejará si el requisito de usuario ó de restricción depende de algún otro.

4.2.1. Requisitos funcionales

Tabla 4.14. Requisito RF-001.

IDENTIFICADOR	RF-001		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Inestable		
ESTADO	Implementado		
DESCRIPCIÓN	Crear una interfaz de usuario capaz de trabajar con redes bayesianas dinámicas permitiendo la visualización de la red y la manipulación de la misma.		
DEPENDENCIAS	RF-002		

Tabla 4.15. Requisito RF-002.

IDENTIFICADOR	RF-002		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La interfaz permitirá el uso simultaneo de la aplicación Netica para la instanciación de variables por parte del uso manual del usuario además de servir como mecanismo de visualización gráfica		
DEPENDENCIAS	RF-007		

Tabla 4.16. Requisito RF-003.

IDENTIFICADOR	RF-003		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El uso de la interfaz tendrá dos funcionamientos diferenciados, el primero será un proceso manual en el cuál es el usuario quien realiza la simulación de las evidencias a través de su participación en el proceso de inferencia en los cortes de tiempo y el segundo es un proceso automático que hace uso de las evidencias generadas por otro sistema creando una simulación de un sistema experto en la que el usuario no interactuará		
DEPENDENCIAS	RF-007 RF-009		

Tabla 4.17. Requisito RF-004.

IDENTIFICADOR	RF-004		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El usuario podrá seleccionar y deseleccionar las relaciones temporales que existan en la red bayesiana dinámica pudiendo ser modificadas antes del inicio de la aplicación		
DEPENDENCIAS			

Tabla 4.18. Requisito RF-005.

IDENTIFICADOR	RF-005		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Recomendable	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Inestable		
ESTADO	Implementado		
DESCRIPCIÓN	Es recomendable proteger al usuario de posibles errores que se puedan realizar en la ejecución de la interfaz, siendo en dicho caso necesaria la notificación del error		
DEPENDENCIAS			

Tabla 4.19. Requisito RF-006.

IDENTIFICADOR	RF-006		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El usuario podrá seleccionar y deseleccionar las relaciones variables de observación existentes en la red bayesiana pudiendo ser modificadas antes del inicio de la aplicación		
DEPENDENCIAS			

Tabla 4.20. Requisito RF-007.

IDENTIFICADOR	RF-007		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La simulación manual se realizará tras la incorporación de una red bayesiana y la notificación de las relaciones temporales participantes en la simulación		
DEPENDENCIAS	RF-004 RF-006		

Tabla 4.21. Requisito RF-008.

IDENTIFICADOR	RF-008		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Recomendable	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Inestable		
DESCRIPCIÓN	La realización de alguna tarea bloqueante por parte del sistema, que requiera de un tiempo de cómputo, será visualizado por el usuario a través de una barra de proceso que vaya señalando cuanto queda hasta su completitud		
DEPENDENCIAS			

Tabla 4.22. Requisito RF-009.

IDENTIFICADOR	RF-009		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La simulación automática se habilitará tras la incorporación de un fichero de entrada de datos, la incorporación de la red bayesiana partícipe y la notificación de las relaciones temporales necesarias en la simulación		
DEPENDENCIAS	RF-004 RF-006 RF-010 RR-001		

Tabla 4.23. Requisito RF-010.

IDENTIFICADOR	RF-010		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La frecuencia de lectura del fichero de datos en la simulación automática, podrá ser modificada antes del inicio de dicho modo de ejecución		
DEPENDENCIAS			

Tabla 4.24. Requisito RF-011.

IDENTIFICADOR	RF-011		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La detención del modo de ejecución manual de la interfaz regresará al estado de inicio notificando del cambio introducido en la red bayesiana en el caso de haber sido modificada		
DEPENDENCIAS			

Tabla 4.25. Requisito RF-012.

IDENTIFICADOR	RF-012		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Recomendable	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Inestable		
ESTADO	Implementado		
DESCRIPCIÓN	La información que no pueda ser utilizada en un momento determinado de ejecución de la interfaz, estará deshabilitada para mejorar la visibilidad de las funciones a emplear		
DEPENDENCIAS			

Tabla 4.26. Requisito RF-013.

IDENTIFICADOR	RF-013		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Inestable		
ESTADO	Implementado		
DESCRIPCIÓN	La aplicación dispondrá de un algoritmo capaz de calcular y actualizar las probabilidades a priori y condicionales de las variables que simule el proceso de inferencia de las redes bayesianas dinámicas		
DEPENDENCIAS			

Tabla 4.27. Requisito RF-014.

IDENTIFICADOR	RF-014		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El usuario dispondrá de un control capaz de realizar un corte de tiempo en el proceso de inferencia de acuerdo con las condiciones actuales encontradas o señaladas en la red bayesiana		
DEPENDENCIAS	RF-013		

Tabla 4.28. Requisito RF-015.

IDENTIFICADOR	RF-015		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Recomendable	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La interfaz permitirá la detención del modo de simulación automático, eliminado el uso del fichero de datos y de la red bayesiana para retomar el estado inicial de la interfaz		
DEPENDENCIAS			

Tabla 4.29. Requisito RF-016.

IDENTIFICADOR	RF-016		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El usuario dispondrá de un control capaz de realizar la simulación de varios cortes de tiempos consecutivos que permitan observar la evolución de la red bayesiana a lo largo del tiempo sin la percepción de ninguna evidencia en la red bayesiana.		
DEPENDENCIAS	RF-013 RF-014		

Tabla 4.30. Requisito RF-017.

IDENTIFICADOR	RF-017		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Opcional	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La interfaz debe permitir la representación gráfica de las variables de observación a lo largo del tiempo de simulación, pudiendo salvaguardar el estado gráfico en cualquier instante de tiempo		
DEPENDENCIAS	RR-004		

Tabla 4.31. Requisito RF-018.

IDENTIFICADOR	RF-018		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El usuario podrá detener tanto el proceso automático como la simulación de un número elevado de cortes de tiempo, deteniendo la tarea invocadora y mostrando los valores calculado hasta el momento		
DEPENDENCIAS			

Tabla 4.32. Requisito RF-019.

IDENTIFICADOR	RF-019		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La interfaz permitirá la reanudación del modo automático de simulación, con lo que el proceso de ejecución se retomará desde el punto en el que se encuentre la aplicación		
DEPENDENCIAS			

Tabla 4.33. Requisito RF-020.

IDENTIFICADOR	RF-019		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	La aplicación debe de ser capaz de interpretar los eventos del usuario y las acciones realizadas en la interfaz Netica para poder llevar a cabo la reproducción de los resultados obtenidos		
DEPENDENCIAS			

4.2.2. Requisitos de restricción

Tabla 4.34. Requisito RR-001.

IDENTIFICADOR	RR-001		
PRIORIDAD	Alta	**FUENTE**	Desarrollador
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El fichero de entrada para el uso automático de la interfaz tiene que seguir un formato predefinido que constará de: ▪ [Nombre de la variable de observación][espacio][Nombre del valor de la variable] Ejemplo: Lluvia1 si Paraguas no		
DEPENDENCIAS			

Tabla 4.35. Requisito RR-002.

IDENTIFICADOR	RR-002		
PRIORIDAD	Alta	**FUENTE**	Desarrollador
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	En la construcción de la red bayesiana dinámica el usuario empleará la aplicación Netica, teniendo la red el formato dependiente de dicho sistema y por tanto la interfaz empleará el mismo formato de almacenamiento		
DEPENDENCIAS			

Tabla 4.36. Requisito RR-003.

IDENTIFICADOR	RR-003		
PRIORIDAD	Alta	**FUENTE**	Cliente
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	Las redes bayesianas dinámicas solamente podrán contener variables discretas, siendo necesario la realización previa de discretización de la variable, si ésta fuese continua		
DEPENDENCIAS			

Tabla 4.37. Requisito RR-004.

IDENTIFICADOR	RR-004		
PRIORIDAD	Alta	**FUENTE**	Desarrollador
AUTOR	Daniel García	**FECHA**	02/10/2009
NECESIDAD	Esencial	**VERIFICABILIDAD**	Alta
ESTABILIDAD	Estable		
ESTADO	Implementado		
DESCRIPCIÓN	El formato del archivo para la representación gráfica de las variables de observación tendrá como valores en los ejes de coordenadas y en el abscisa, la probabilidad y el tiempo , respectivamente.		
DEPENDENCIAS			

Capítulo V. Desarrollo de la solución

Una componente importante en la construcción de esta memoria, corresponde al desarrollo del software que permita reproducir el proceso de inferencia de las redes bayesianas dinámicas a través del desarrollo de una interfaz que implemente el algoritmo necesario para su realización.

Este capítulo cumple dicha función, recogiendo el proceso de desarrollo de la interfaz, presentando la metodología empleada, el diseño conceptual, la descripción de implementación así como otros aspectos que satisfacen las necesidades planteadas en los objetivos iniciales de este proyecto.

5.1. Metodología

El proceso de desarrollo de software que se ha seguido para la construcción de la herramienta software ha sido el modelo denominado desarrollo evolutivo de tipo exploratorio. La idea de este modelo consiste, en el desarrollo de una implementación del sistema inicial, exponerla a los comentarios del usuario e ir refinándola en un número sucesivo de versiones hasta que se obtenga el sistema adecuado. En la figura 5.1., se presenta una visión gráfica del modelo evolutivo donde las actividades concurrentes, se realizan durante el desarrollo de las versiones hasta llegar al producto final.

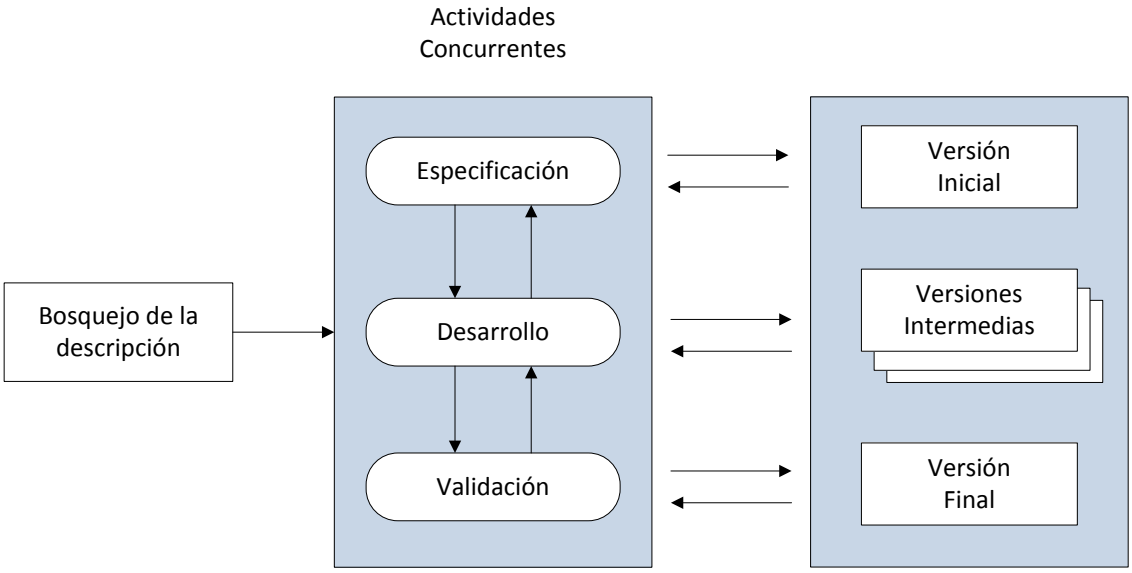

Figura 5.1. Modelo de desarrollo evolutivo.

El objetivo de este desarrollo es explotar a través del usuario los requisitos hasta llegar a una versión final. El desarrollo se empieza con aquellas partes que se tienen más claras y cada vez se irá evolucionando conforme se añadan nuevas características propuestas por el usuario. Cuya principal ventaja es que la especificación se puede desarrollar de forma creciente.

5.2. Diseño arquitectónico

El diseño de la aplicación desarrollada se compone de tres componentes que nos permite encapsular la lógica de negocio en entidades independientes que se comunican entre sí a través de pasos de mensajes. De que tal modo que, el desarrollo se ha basado en el modelo arquitectónico MVC (Modelo-Vista-Controlador) al ser el modelo que mejor se acopla a los requerimientos de la aplicación. Los componentes del MVC se pueden describir en:

- El **Modelo**, contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
- La **Vista** o interfaz de usuario, presenta el modelo en un formato adecuado además de contener los mecanismos de interacción con éste.
- El **Controlador**, gestionará los mensajes recibidos del usuario (interfaz de entrada) y los traduce a mensaje comprensibles por el modelo conceptual; es responsable, también del flujo de la aplicación.

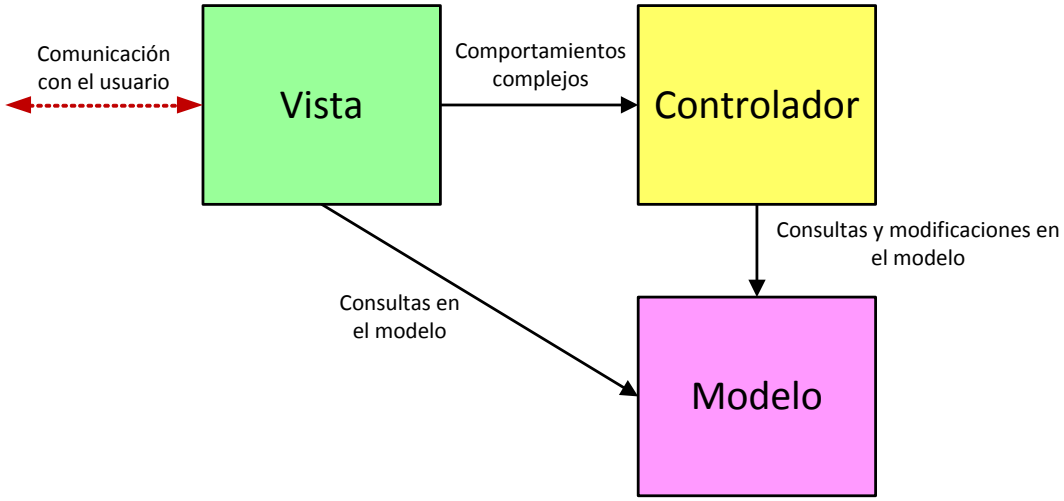

Figura 5.2. Arquitectura modelo-vista-controlador.

Como consecuencia del empleo de dicho modelo arquitectónico se pueden citar las siguientes ventajas, que principalmente se pueden resumir, en que permiten la adición de nuevas funciones, mejoras o futuras ampliaciones al entorno de usuario por parte de otros desarrolladores no partícipes de la primera versión inicial. Las ventajas son:

- Es un modelo fácil de explicar y entender.
- Reduce dependencias y potencia la reutilización.
- Permite dividir el trabajo en base a distintos roles (el diseñador gráfico no tiene por qué saber cómo se ha implementado el resto de componentes).
- Simplicidad en el mantenimiento de los sistemas.
- Facilidad para desarrollar prototipos rápidos.

Pero en cambio, también se presentan otros inconvenientes como son:

- Posee una mayor complejidad y por tanto mayor coste (sobre todo en aprendizaje para los nuevos desarrolladores) que otros modelos más simples.
- La distribución en componentes obliga a crear y mantener un mayor número de ficheros.

- El ceñimiento a una estructura predefinida, puede ocasionar incrementos de complejidad del sistema, ya que existen problemas que son más difíciles de resolver respetando el patrón MVC que el empleando otros modelos arquitectónicos.

En nuestro caso particular de desarrollo del software para el tratamiento de redes bayesiana dinámicas, el componente referente a la vista está basada en formulario de *Windows Form*.
Windows Forms es una plataforma de desarrollo de aplicaciones para Microsoft Windows, basada en .NET Framework. Este marco de trabajo proporciona un conjunto de clases claro, orientado a objetos y ampliable, que permite desarrollar complejas aplicaciones para Windows. Además de que estos *Windows Form* pueden actuar como interfaz de usuario local en una solución distribuida de varios niveles.

El componente controlador estará representado por su clase correspondiente que interpretará los mensajes recibidos por la vista y realizará las acciones necesarias en el modelo provocando cambios de estado en la representación interna de los datos, así como en su visualización. Además, contendrá los algoritmos y el flujo de información necesarios para realizar la simulación de inferencia de las redes bayesiana dinámicas.

Por último, la descripción del modelo vendrá representada por el componente de software *Netica 4.09 Object Library* desarrollada por *Norsys Software Corporation*, que nos permitirá trabajar con la aplicación Netica. La incorporación de la referencia COM al proyecto se puede consultar en el anexo B: Manual de instalación. Este componente será el encargado de recibir las peticiones enviadas por el controlador, procesarlas y devolver al controlador la información resultante para que sea presentado al usuario a través de la capa de vista.

La figura 5.3. presenta el diagrama de componentes del sistema antes de entrar detallar la descomposición en módulos.

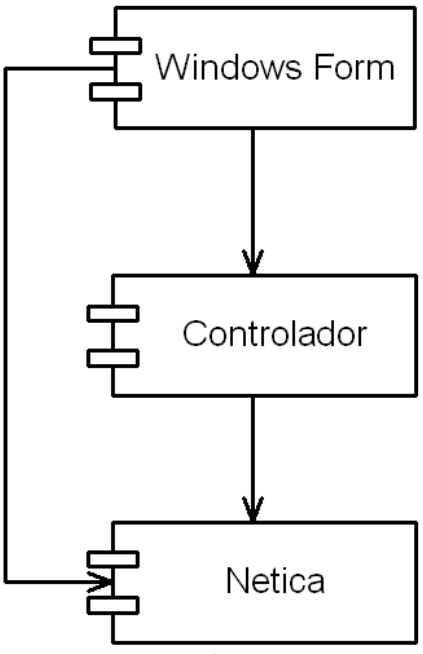

Figura 5.3. Diagrama de componentes.

5.3. Descomposición en módulos

En este apartado se detallarán cada uno de los módulos de los que está compuesto el sistema

5.3.1. Vista

Como ya se presentó en el apartado anterior, éste módulo será el encargado de interactuar con el usuario y de presentarle la información.

La clase "Form" será la responsable de contener la declaración y el tratamiento de los eventos de todos los objetos de Windows Forms. La clase "PFC" será la encargada de lanzar la ejecución del sistema a través de la función "main".

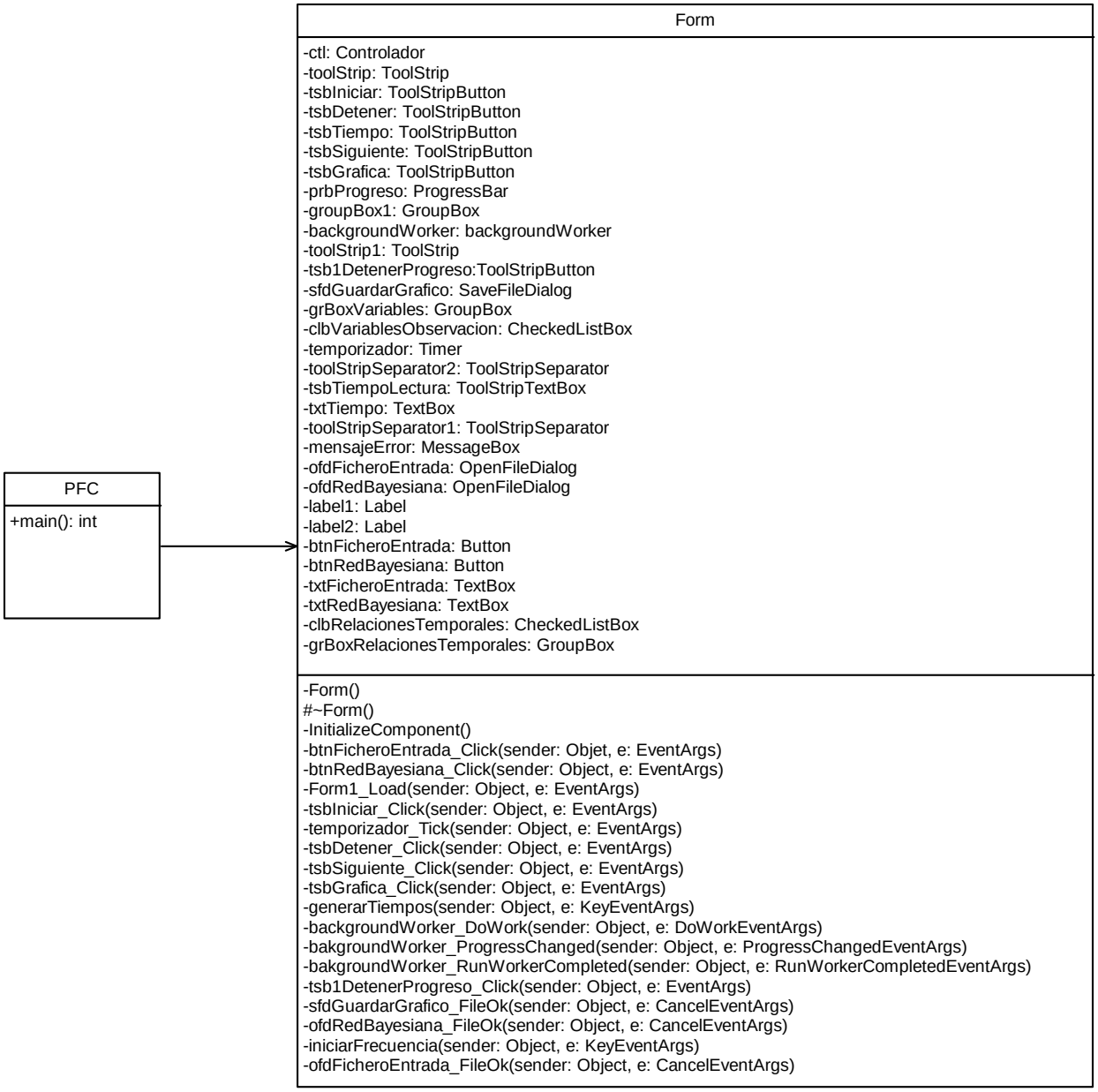

Figura 5.4. Diagrama de clase de la Vista.

La descripción de los atributos de la clase “Form”, no se ve necesaria debido a que correspondería a especificar el diseño de la interfaz la cual se puede consultar en el Anexo C: Manual de usuario dentro del apartado Interfaz desarrollada.

5.3.1.1. Funciones

A continuación se describirán los objetivos de las funciones de la clase “Form”.

- btnFicheroEntrada_Click: evento que muestra el dialogo para la entrada del fichero de datos.
- btnRedBayesiana_Click: evento que muestra el dialogo para la entrada de la red bayesiana.
- Form1_Load: carga el formulario en memoria pero sin visualizarse.
- tsbIniciar_Click: evento que inicia el modo automático o manual (dependiendo del estado de la interfaz) a través del controlador.
- temporizador_Tick: evento que realiza llamadas al controlador para ejecutar la lectura del fichero de datos cada tick de reloj.
- tsbDetener_Click: evento que detiene la ejecución del sistema devolviendo la interfaz a su estado inicial.
- tsbSiguiente_Click: evento que realiza la simulación de un corte de tiempo a través del controlador.
- tsbGrafica_Click: evento que se encarga de mostrar el diálogo de salida para el fichero que contiene el gráfico en Gnuplot.
- generarTiempos: evento que comprobará que el valor introducido en la caja de texto de generar tiempos es correcto.
- backgroundWorker_DoWork: evento que ejecuta de manera asíncrona la función del controlador que se encarga de calcular varios cortes de tiempo.
- bakgroundWorker_ProgressChanged: evento que se dispara cada vez ha habido un progreso en la actividad de calcular varios cortes de tiempo.
- bakgroundWorker_RunWorkerCompleted: evento que se dispara automáticamente cuando la tarea del DoWork ha finalizado.
- tsb1DetenerProgreso_Click: evento que detiene la ejecución de varios cortes de tiempo cuando la interfaz se encuentra en modo manual. Si la interfaz se encuentra en modo automático, inicia o detiene la simulación automática.
- sfdGuardarGrafico_FileOk: evento encargado de guardar el gráfico al validar el usuario el nombre del fichero.
- ofdRedBayesiana_FileOk: evento que carga la red bayesiana en la interfaz a través del controlador tras seleccionar el usuario la ubicación de la red bayesiana.
- iniciarFrecuencia: evento que comprobará que el valor introducido en la caja de texto de frecuencia de lectura del fichero de datos es correcto.
- ofdFicheroEntrada_FileOk: evento que carga el fichero de datos en la interfaz a través del controlador tras seleccionar el usuario la ubicación del fichero de datos.

5.3.2. Controlador

Este módulo está compuesto por la clase "Controlador" que se encargará de gestionar las llamadas del usuario al modelo. También contendrá los algoritmos que permitirán realizar la simulación del proceso de inferencia de las redes bayesianas dinámicas.

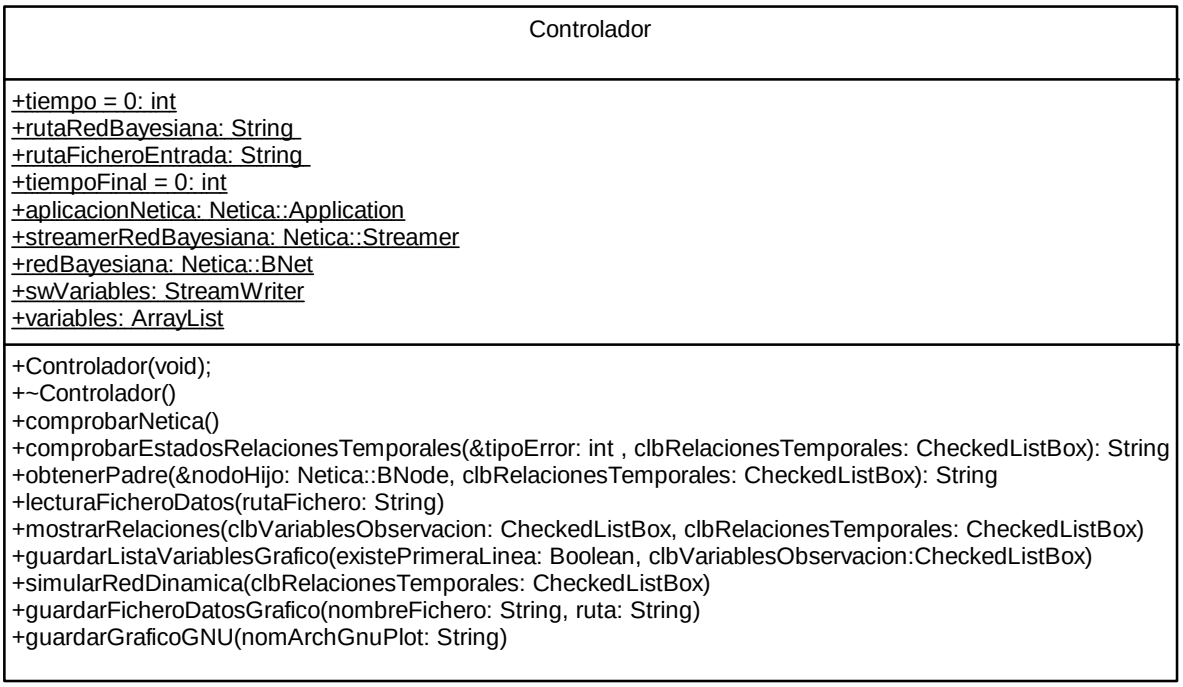

Figura 5.5. Diagrama de clase del Controlador.

5.3.2.1. Atributos

A continuación se describen los atributos que definen a la clase "Controlador":

- Tiempo = 0: int → contiene el valor actual del número de cortes de tiempo realizados.
- rutaRedBayesiana: String→ almacena la dirección de la ubicación física donde se encuentra la red bayesiana
- rutaFicheroEntrada: String→ almacena la dirección de la ubicación física donde se encuentra el fichero de entrada de datos.
- tiempoFinal: int→ contiene el valor final del número de cortes de tiempo a realizar.
- aplicacionNetica: Netica::Application→Objeto de la clase Netica::Application necesaria para poder manejar la aplicación Netica.
- streamerRedBayesiana: Netica::Streamer→ Objeto de la clase Netica::Streamer necesario para el flujo de datos entre la aplicación Netica y el controlador.
- redBayesiana: Netica::Bnet→ Objeto de la clase Netica::Bnet que permite trabajar con la clase BNet que implementa la red bayesiana.
- swVariables: StreamWriter→ Variable que nos permitirá crear un flujo de salida hacia un fichero para los ficheros de gráficos.
- variables: ArrayList → estructura de tipo lista de arrays dinámica que se usa para ir almacenando las probabilidades de las evidencias de la red para las variables observadas.

5.3.2.2. Funciones

A continuación se describirán el objetivo de las funciones de la clase “Controlador”.

- comprobarNetica→ comprueba si la aplicación Netica se encuentra abierta o ha sido cerrada por el usuario. En caso de estar cerrada instancia los objetos necesarios del componente Modelo para que se puedan usar.
- comprobarEstadosRelacionesTemporales→ este método comprobará requisitos previos sobre el estado de las relaciones temporales para que se pueda realizar una ejecución correcta del proceso de inferencia. Por un lado realiza la comprobación de que al menos una relación temporal haya sido seleccionada y por otro verifica que los estados entre la variable padre su hijo de una relación temporal se correspondan.
- obtenerPadre→ devuelve el nombre del nodo padre del nodo hijo pasado como parámetro. En caso de no encontrarse devolverá una cadena vacía.
- lecturaFicheroDatos→este método se encarga de leer del fichero de entrada de datos las evidencias de las variables y hacerlas visibles en la aplicación Netica.
- mostrarRelaciones→es el método responsable de la inicialización de la aplicación Netica así como del Streamer y del objeto Bnet necesarios para la gestión del componente Modelo. Su función también consistirá en mostrar las relaciones temporales y las variables de observación en la Vista.
- guardarListaVariablesGrafico→ obtiene la información de las variables de observación preseleccionadas por el usuario insertando las probabilidades actuales de sus estados en un fichero temporal para su posterior uso a través de la opción habilitada al usuario para salvaguardar los resultados en un fichero gráfico.
- simularRedDinamica→ este método es el responsable de la simulación del proceso de inferencia de una red bayesiana dinámica presentado en la Tabla 3.1. Proceso de actualización de una RBD.
- guardarFicheroDatosGrafico→ su función consiste en realizar el cambio del carácter “,” por “.” de los resultados numéricos correspondientes al fichero de datos asociado al archivo con extensión “.plz” (ver apartado 5.4.3 Ficheros de gráfico)
- guardarGraficoGNU→ la función de este método consiste en guardar el fichero con extensión “.plz”, con las instrucciones necesarias para que este sea visualizado en el programa Gnuplot. También se encargará realizar la llamada necesaria para almacenar

5.3.3. Modelo

El componente modelo recoge la descripción de las clases de *Netica 4.09 Object Library* que se han utilizado para el desarrollo de la aplicación así como la descripción de los métodos y atributos empleados. Una especificación más detallada de las clases, método y atributos que contiene esta biblioteca puede obtenerse del examinador de objetos de Visual Studio o consultado con la API ofrecida por Norsys (52).

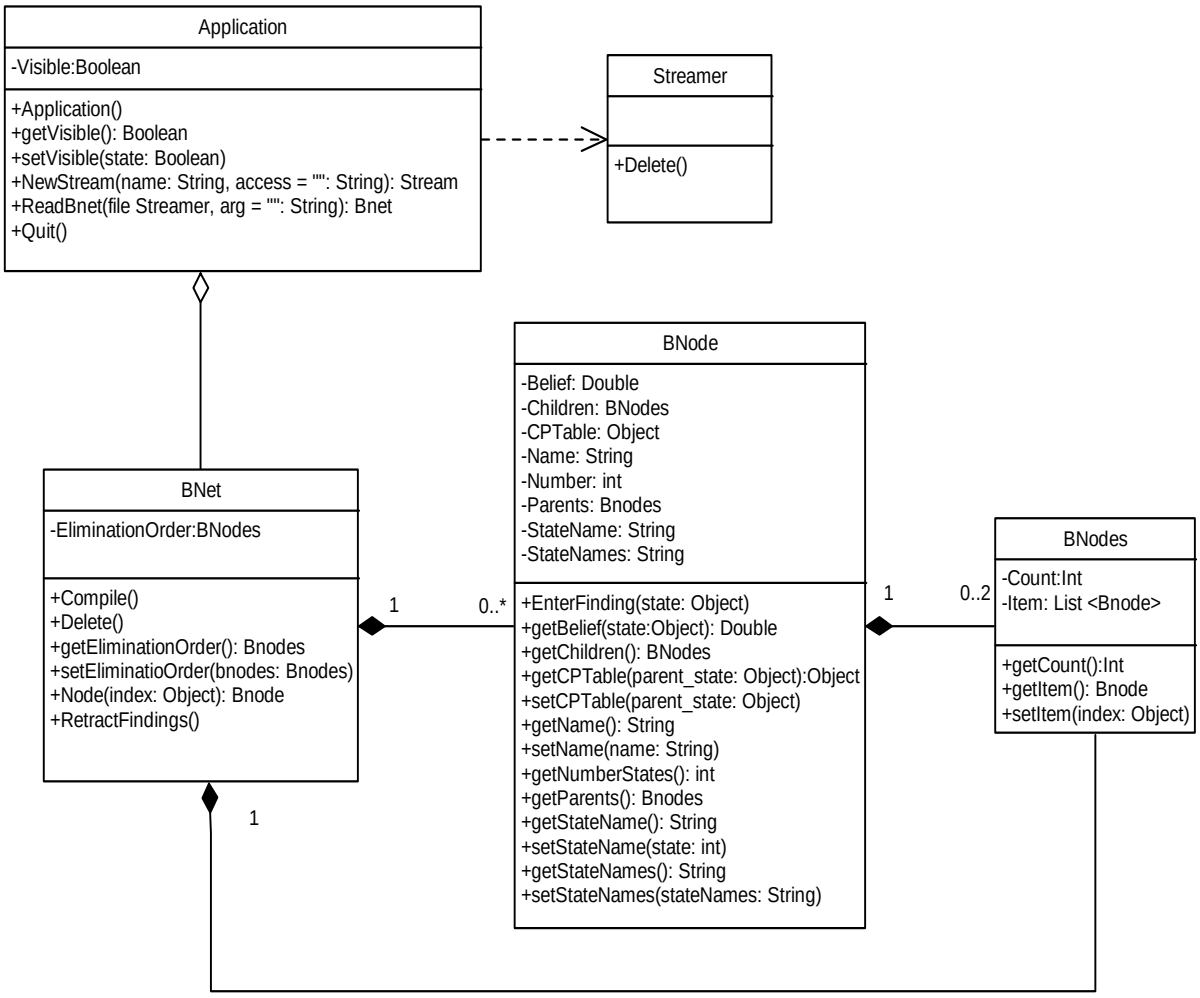

Figura 5.6. Diagrama de clase del Modelo.

Antes de describir los atributos y las funciones empleadas en el proceso de implementación del proyecto y recogidas en el diagrama de clase anterior, se realizará una descripción del significado de cada clase y de su objetivo.

- La clase "Application" representa la abstracción de la aplicación Netica. Su uso es necesario para el inicio de la aplicación y la de gestión de todas las opciones que se puedan realizar en la interfaz de usuario de Netica.
- La clase "Streamer" representa el flujo de datos que nos permite leer/escribir las redes bayesianas desde la aplicación.
- La clase "BNet" representa la abstracción en el mundo real de una red bayesiana. Su función consiste en permitir realizar acciones y obtener

información sobre el estado de la red bayesiana completa. Esta clase nos permitirá realizar la propagación de las evidencias a través de la red.

- La clase "BNode" representa la abstracción en el mundo real de una variable (o nodo) de una red bayesiana. Contendrá toda aquella información perteneciente a sus estados como pueden ser tipo, los nombres y las tablas probabilidades, etc. además de contener las relaciones entre sus variables padres y sus variables hijos.
- La clase "BNodes" es un contenedor de objetos de tipo "BNode" que nos permitirán almacenar información sobre las relaciones del nodo, como quiénes son sus padres y sus hijos.

5.3.3.1. Atributos clase Application

- Visible: Boolean→ atributo que almacenará el estado de visibilidad de cara al usuario de la aplicación Netica.

5.3.3.2. Funciones clase Application

- getVisible→ devuelve el estado en el que se encuentra el atributo Visible.
- setVisible→ permite modificar el estado de visibilidad del objeto Application
- NewStream→ permite crear un nuevo flujo de datos entre el archivo pasado como parámetro y la aplicación Netica.
- ReadBnet→ realiza la instanciación de los distintos objetos que se encuentren tras la lectura del fichero de almacenamiento que contiene la definición de la red bayesiana.
- Quit→ cierra la aplicación Netica.

5.3.3.3. Funciones clase Streamer

- delete→ elimina el objeto instanciado liberando todos los recursos contenidos en memoria por su uso.

5.3.3.4. Atributos clase BNet

- EliminationOrder:BNodes→es el atributo que contiene la lista ordenada de todos los nodos de la red bayesiana(excepto constantes y nodos de utilidad), utilizados para guiar al proceso de complicación.

5.3.3.5. Funciones clase BNet

- Compile→realiza el proceso de compilación de una red bayesiana por el cual se realiza el proceso de actualización de las probabilidades de las variables acorde al estado de la red.
- Delete→ elimina el objeto instanciado liberando todos los recursos contenidos en memoria por su uso.
- getEliminationOrder→ devuelve la lista ordenada de nodos almacenada en el atributo EliminationOrder.

- setEliminatioOrder→ permite modificar la lista ordenada de nodos almacenada en el atributo ElimitionOrder.
- Node→recupera un objeto de tipo "BNode" contenido en la red bayesiana a través del parámetro pasado como argumento que puede ser el nombre o el índice de un número entero (en cuyo caso 0 indica el primer nodo).
- RetractFindings→ elimina todas las evidencias existentes en la red bayesiana de todos los nodos.

5.3.3.6. Atributos clase BNode

- Belief: Double → contiene la creencia actual de cada estado del nodo
- Children: BNodes→ atributo que contiene una lista de "Bnodes" con los hijos del nodo.
- CPTable: Object→ contiene la probabilidad condicional del nodo.
- Name: String→ almacena el nombre del nodo.
- Number: int→ valor numérico que representa en número de estados del nodo.
- Parents: Bnodes → atributo que contiene una lista de "Bnodes" con los padres del nodo.
- StateName: String → atributo que contiene el nombre de un estado concreto del nodo.
- StateNames: String→atributo que contiene todos los nombres de los estados del nodo separados por comas.

5.3.3.7. Funciones clase BNode

- EnterFinding→ permite introducir un hallazgo de este nodo. Es un método sobrecargado ya que permite la introducción del hallazgo a través de un entero un booleano o un "String".
- getBelief→ obtiene la creencia actual de cada estado del nodo teniendo en cuenta los resultados actuales de la red.
- getChildren → obtiene una lista de "BNode" con los hijos del nodo
- getCPTable → devuelve la probabilidad condicional o a priori del nodo dada los estados en los que se encuentran sus padres.
- setCPTable→ permite especificar la probabilidad condicional o a prior.
- getName→ devuelve el nombre del nodo.
- setName→permite establecer un nuevo nombre al nodo.
- getNumberStates→ devuelve el entero con el número de estados que contiene el nodo.
- getParents→ obtiene una lista de "BNode" con los padres del nodo.
- getStateName→devuelve el nombre del estado a través de un índice pasado como argumento que indica la posición del estado del nodo.
- setStateName→ modifica el nombre del estado a través del índice pasado como argumento que indica la posición del estado del nodo.
- getStateNames→ devuelve en una cadena de caracteres los nombres de los estados de este nodo separados por comas.
- setStateNames→ permite establecer los estados del nodo a través de una cadena de caracteres en la que los nombres tiene que ir separados por comas.

5.3.3.8. Atributos clase BNodes

- Count:Int → valor numérico que hace referencia al número de elementos que contiene el atributo "Item".
- Item: List <Bnode>→ atributo que representa a la lista de objetos de tipo "BNode".

5.3.3.9. Funciones clase BNodes

- getCount→ devuelve el número de elementos que contiene el objeto de tipo "BNodes" dentro del atributo "Item"
- getItem→ permite obtener un elemento de la lista de "BNode".
- setItem→permite modificar un elemento de la lista de "BNode".

5.4. Formatos de Entrada y Salida

En este apartado se detallarán los formatos que deben poseer los dos ficheros de entrada de la interfaz desarrollada, así como los dos ficheros de salida que se generan a la hora de guardar un gráfico.

Los dos tipos de archivos que la interfaz necesita como entrada, corresponden respectivamente, al denominado como "Fichero de datos" y al denominado como "Red bayesiana".

Los otros dos ficheros que se generan de salida corresponden al archivo de órdenes necesarias para representar un gráfico en Gnuplot y el correspondiente al anterior que contiene los datos necesarios para la representación.

5.4.1. Fichero de datos

Este archivo contendrá las evidencias de las variables que se quieran inferir en el modo automático de la interfaz. Su extensión es ".drb" (datos red bayesiana) y consiste en un texto plano en el que cada línea del fichero constará de [Nombre de la variable][ ][Nombre del estado X evidenciado] de la red bayesiana que vayan a representar. La imagen 5.1. presenta un ejemplo visual del fichero de entrada correspondiente a la red bayesiana del caso práctico 2. Los atributos que contendrá este fichero serán las variables de observación que se presenten en la red bayesiana dinámica, las cuales, se pueden evidenciar para la obtención de las variables de estimación.

```
Valvula1 correctoOk
Valvula2 abiertaR0
DistribucionFluido Si
Valvula3 cerradaRC
```

Imagen 5.1. Ejemplo del fichero de datos.

5.4.2. Red bayesiana

Este archivo representa el formato de almacenamiento y/o cargar de una red bayesiana en la aplicación Netica. Sus extensiones pueden ser ".neta" y ".dne" siendo ambos válidos para la interfaz. El primero se corresponde con el fichero binario de la red bayesiana, el cual, no puede ser visualizado a través del usuario con la ayuda de un editor de textos y el segundo, es la misma versión que el primero pero en texto plano y por tanto visualizable.

La explicación del formato de este archivo de almacenamiento de una red bayesiana puede consultarse en (53), su explicación en este documento no se ve necesaria ya que

la construcción del mismo será a través del programa Netica y por tanto su realización será proporcionada automáticamente a través de su interfaz de usuario.

Un ejemplo del formato de almacenamiento anteriormente comentado, para redes bayesiana, puede ser consultado en imagen que se presenta a continuación.

```
node Lluvia {
    kind = NATURE;
    discrete = TRUE;
    chance = CHANCE;
    states = (si, no);
    parents = ();
    probs =
        // si            no
          (0.5,          0.5);
    whenchanged = 1262715666;
    belief = (0.5, 0.5);
    visual V3 {
        center = (156, 114);
        height = 5;
        };
    };

node Lluvial {
    kind = NATURE;
    discrete = TRUE;
    chance = CHANCE;
    states = (si, no);
    parents = (Lluvia);
    probs =
        // si            no              // Lluvia
          ((0.7,         0.3),           // si
           (0.3,         0.7));          // no      ;
    numcases = 1;
    whenchanged = 1261050349;
    belief = (0.5, 0.5);
    visual V3 {
        center = (342, 114);
        height = 1;
        };
    };
```

Imagen 5.2. Formato de almacenamiento de una red bayesiana.

5.4.3. Ficheros de gráficos

En cuanto a la generación de los gráficos para su visualización posterior en Gnuplot, dos son los archivos que se crearán. El primer archivo contiene los datos necesarios para la representación gráfica y consiste en un texto plano con extensión ".txt" que se nombra a partir del nombre del segundo fichero (que se verá posteriormente) más la adición de la palabra "Datos". El formato de este fichero consta de una primera línea constituida por la descripción del contenido de cada columna, que representan el valor del tiempo y el estado de las variables de observación seleccionadas previamente por el usuario. A partir de aquí, cada nueva línea contendrá los valores numéricos de las variables. La imagen 5.3. representa un ejemplo visual del formato anteriormente explicado.

```
TIEMPO Lluvia=si Lluvia=no
0 0.500000596046448 0.499999433755875
1 1 0
2 0.299999982118607 0.699999988079071
3 0.579999983310699 0.420000016689301
4 0.468000024557114 0.531999945640564
5 0.512799918651581 0.487200051546097
6 0.494880080223084 0.505119919776917
7 0.502047955989838 0.49795201420784
8 0.499180823564529 0.500819206237793
9 0.50032764673233 0.49967235326767
```

Imagen 5.3. Ejemplo salida del fichero de datos gráficos.

En este fichero se presentarán la secuencia de variables estimadas que se hubiesen seleccionado del entorno de usuario y que se quieran explorar y/o visualizar a lo largo del tiempo. Normalmente la presentación de este fichero contendrá la secuencia en el tiempo de las variables de estimación a visualizar que nos permitan realizar la evaluación de la situación.

El segundo de ellos hace referencia al archivo con extensión ".plt" que contendrá las órdenes en Gnuplot como si éstas se estuviesen tecleando dentro de la línea de comandos. El nombre vendrá dado por el usuario al salvaguardar el gráfico.
Este fichero tiene un formato estándar que se genera automáticamente y se corresponderá con las variables de observación seleccionadas por el usuario. En la imagen 5.4. (ver página siguiente) se puede observar un ejemplo del formato mencionado el cuál constará siempre de:

- Líneas estáticas: líneas que no cambian para distintos gráficos.
  - Título del gráfico (línea "*set title "GRAFO TEMPORAL""*)
  - Título del eje de coordenadas (línea "*set xlabel "TIEMPO""*)
  - Título de eje de abscisas (línea "*set ylabel "PROBABILIDAD""*)
  - Pausa para visualización (línea "*pause -1*")
- Líneas dinámicas: líneas que varían según las variables de observación seleccionadas y según el nombre otorgado por el usuario al fichero ".plt" al

guardar el gráfico. En el caso del ejemplo, el comando "*plot "graficoDatos.txt" using 1:2 t "Lluvia=si" with lines,"graficoDatos.txt" using 1:3 t "Lluvia=no" with lines"* representa este tipo de línea donde se puede observar el siguiente formato:

- *plot* [nombre del fichero de datos] *using* 1:[X] *t* "[Nombre estado de varibles=estado]" *witch lines,* [nombre del fichero de datos] *using* 1:[X] *t* "[Nombre estado de varibles=estado]" *witch lines …*

Donde [X] hace mención a la columna de datos del fichero de datos a utilizar, incrementándose este valor para representar todos los estados de las variables.

```
set title "GRAFO TEMPORAL"
set xlabel "TIEMPO"
set ylabel "PROBABILIDAD"
set grid
plot "graficoDatos.txt" using 1:2 t "Lluvia=si" with
lines,"graficoDatos.txt" using 1:3 t "Lluvia=no" with lines
pause -1
```

Imagen 5.4. Formato del fichero ".plt".

Capítulo VI. Experimentación

En este capítulo se describe la experimentación llevada a cabo tras el desarrollo de la aplicación. Se presentarán varios casos prácticos que nos permitan demostrar el funcionamiento correcto del proceso de inferencia implementado. A la vez se explicarán los casos prácticos para aprender de su utilización para futuras experimentaciones en otros dominios.

En los casos prácticos que se presentan en este apartado, se ha omitido tanto el proceso de construcción de la red bayesiana dinámica como el de obtención de sus probabilidades asociadas.

6.1. Primer caso práctico

El objetivo de este primer caso práctico es el de presentar un ejemplo sencillo de resolución de un problema a través del empleo de una red bayesiana dinámica y su posterior ejecución y simulación en la interfaz desarrollada.

Este primer caso práctico está basado en el ejemplo del guarda de seguridad de Russell y Norving (32). El enunciado es el siguiente:
"Suponga que es un guardia de seguridad en alguna instalación subterránea secreta. Quiere saber si está lloviendo hoy, pero su única información del mundo exterior se produce cada mañana cuando ve al director entrando con, o sin, paraguas."

En este caso cada día que pasa, únicamente se tiene la evidencia de observación de si el director trae o no el paraguas siendo ésta nuestra variable de observación, y la variable de predicción, si está lloviendo o no. La red bayesiana asociada al problema se presenta en la imagen 6.1. donde se pueden observar la credibilidad de los estados de las variables tras la propagación inicial, donde no se ha presentado por el momento, ninguna evidencia.

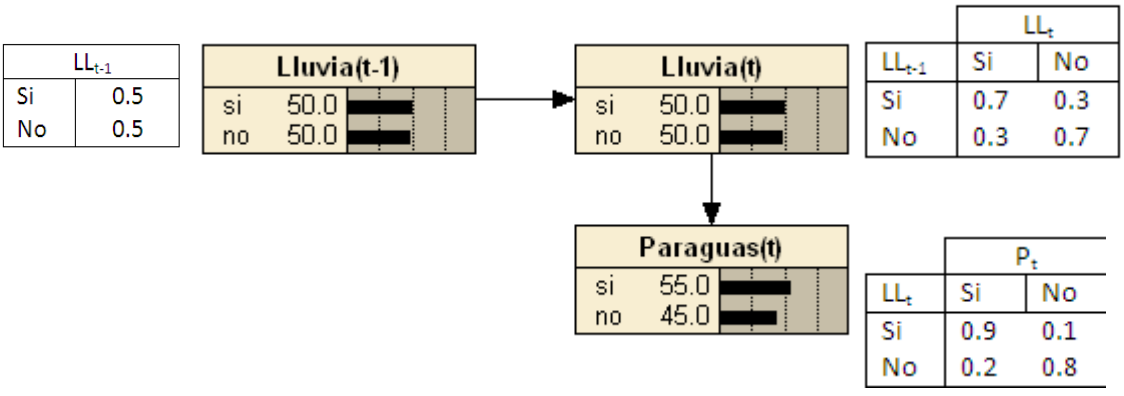

Imagen 6.1. RBD. Caso nº1.

Se denotará como la abreviación "LL_t" a la variable Lluvia en un instante t de tiempo y a con "P_t" a la variable Paraguas. De la observación de la red se obtiene que la relación temporal será entre las variables LL_{t-1} y LL_t tal que $P(LL_t|LL_{t-1})$ y el modelo de observación será entre las variable LL_t y P_t de modo que $P(P_t|LL_t)$.

En cuanto a las probabilidades presentadas al lado de las variables en la imagen 6.1. es de suponer que el guardia antes de entrar en la instalación tenía una creencia a priori sobre si llovió el día 0, justo antes de que comience la secuencia de observaciones.

- Si en el día 1 (t = 1), se observa que el paraguas está presente, P_1 = si, la predicción desde t = 0 hasta t = 1 es
  $P(LL_1) = \sum_{LL_0} P(LL_1|LL_0)P(LL_0)P(LL_1) = \langle 0.7; 0.3\rangle * 0.5 + \langle 0.3; 0.7\rangle * 0.5 = (0.5; 0.5)$,
  que actualizándola con la evidencia para t = 1 produce
  $P(LL_1|P_1) = \alpha\, P(P_1|LL_1)P(LL_1) = \alpha\langle 0.9; 0.2\rangle\langle 0.5; 0.5\rangle = \alpha\langle 0.45; 0.1\rangle \approx \langle 0.818; 0.182\rangle$.
  Como se puede observar en la imagen 6.2. el resultado numérico obtenido se corresponde con el valor gráfico obtenido tras la incorporación de la evidencia en la red bayesiana.

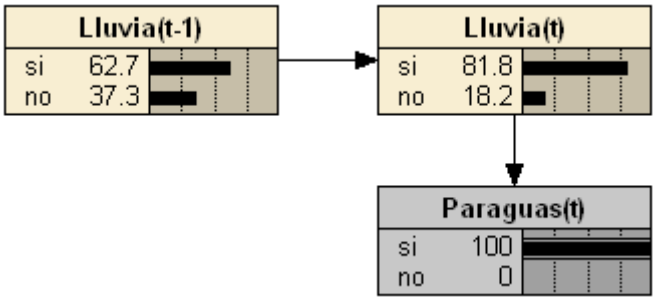

Imagen 6.2. Estado de la red tras el primer corte de tiempo. Caso nº1.

- Si en el día 2 (t = 2), se observa que el paraguas está presente, P_2= si, la predicción desde t = 1 hasta t = 2 es
  $P(LL_2|P_1) = \sum_{LL_1} P(LL_2|LL_1)P(LL_1|P_1)\, P(LL_2) = \langle 0.7; 0.3\rangle * 0.818 + \langle 0.3; 0.7\rangle * 0.182 \approx \langle 0.627; 0.373\rangle$
  que actualizándola con la evidencia para t = 2 produce
  $P(LL_2|P_1, P_2) = \alpha\, P(P_2|LL_2)P(P_1|LL_1) = \alpha\langle 0.9; 0.2\rangle\langle 0.627; 0.373\rangle = \alpha\langle 0.565; 0.075\rangle \approx \langle 0.883; 0.117\rangle$.
  Que como se puede apreciar en la imagen 6.3. se vuelve a coincidir con resultado gráfico obtenido tras la incorporación de la evidencia y la simulación de un corte de tiempo en la aplicación.

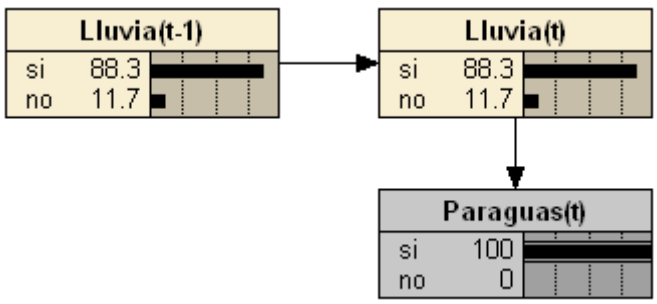

Imagen 6.3. estado de la red tras el segunco corte de tiempo. Caso nº1.

Como se puede observar de la simulación anterior, la probabilidad de lluvia aumenta del día 1 al día 2 porque la lluvia persiste. Aunque se piense que la probabilidad irá en aumento según pasen los días y, se siga observando la presencia del paraguas, esta intuición nos llevaría a cometer un error. Esto es debido a que se obtendrá un punto máximo para la variable Luvia del cuál por más que se tenga la certeza de la presencia del paraguas no se puede mejor la predicción de lluvia debido a la incertidumbre inicial introducida en las probabilidades condicionales de la variable.

La imagen 6.4. presenta el grafo temporal de las variables Lluvia y Paraguas del ejemplo anterior tras la realización de ocho cortes de tiempo donde la variable de observación (paraguas) siempre estaba presente. Se puede verificar por un lado los valores obtenido para desde t = 0 hasta t = 2 para $P(LL_2|P_1, P_2)$ y $P(LL_2|P_1)$; y por otro para verificar el punto máximo que se alcanza para la variable Lluvia tras un periodo corte de tiempo.

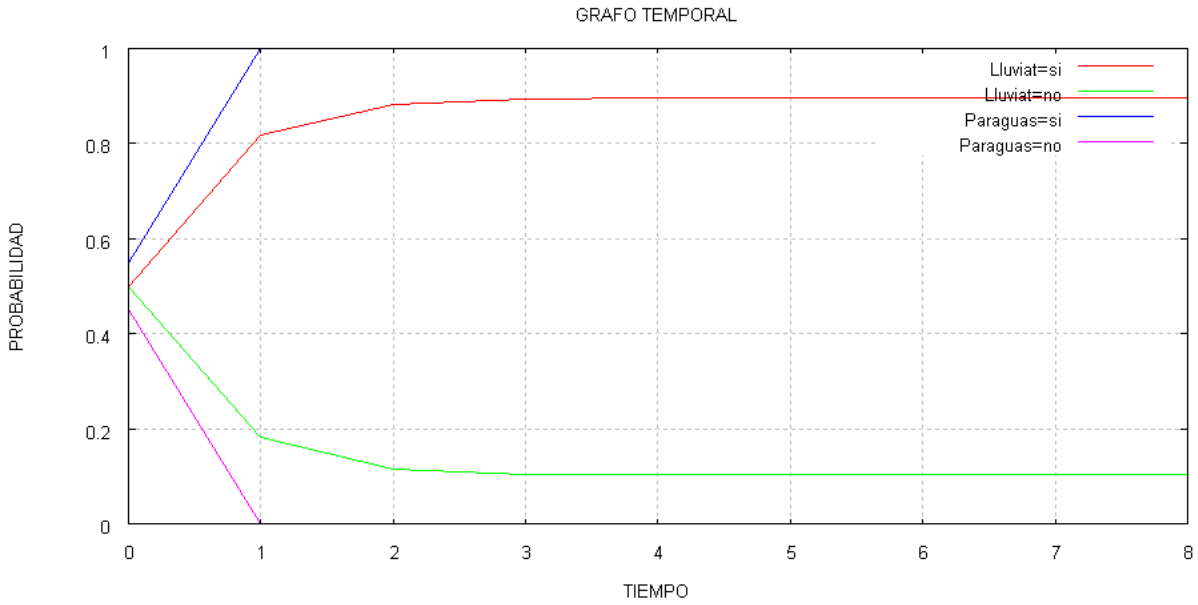

Imagen 6.4. Grafo temporal originado. Caso nº1.

6.2. Segundo caso práctico

Este segundo caso práctico presenta la evolución de un sistema compuesto por tres válvulas que controlan el flujo de un fluido (54). La siguiente figura presenta la colocación de las válvulas en el suministro de fluido.

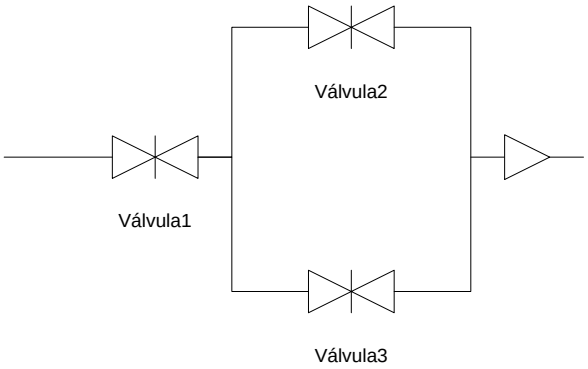

Figura 6.1. Caso práctico nº2.

En este sistema dinámico, cada válvula presenta tres posibles estados de los cuales dos son modos erróneos ya que ocasionan un funcionamiento incorrecto en la distribución del fluido. Los estados son:

- OK: las funciones de la válvula funcionan con normalidad.
- VA: la válvula siempre permanece abierta (modo erróneo).
- VC: la válvula permanece cerrada siempre (modo erróneo).

El objetivo consiste en ser capaces de predecir si el sistema sigue siendo controlable a lo largo de tiempo y por tanto se mantiene el suministro del fluido. Por ejemplo:

- Si las válvulas 2 y 3 permanecen cerradas entonces el sistema no permitirá el flujo de fluidos.
- Si las válvulas 2 y 3 permanecen abiertas, entonces el paso de fluido puede ser controlado a través de la válvula 1 (siempre que esté funcionando correctamente).

Este ejemplo es muy representativo ya que en la mayoría de los sistemas industriales, mecánicos, electrónicos, etc., los componentes de los sistemas tienen una vida útil limitada, que tras el paso del tiempo, empiezan a deteriorarse y por consiguiente dejan de funcionar correctamente. Además es normal que incluso cuando éstos son nuevos presentan una tasa de fallo aunque de por sí suela ser muy pequeña.
En nuestro caso las tasas error de cada válvula es el siguiente:

- Válvula 1
  - Se queda siempre abierta $P(V_1 = abierta) = 1 * 10^{-3}$
  - Se queda siempre cerrada $P(V_1 = cerrada) = 2 * 10^{-3}$
- Válvula 2:
  - Se queda siempre abierta $P(V_2 = abierta) = 2 * 10^{-3}$
  - Se queda siempre cerrada $P(V_1 = cerrada) = 3 * 10^{-3}$
- Válvula 3:
  - Se queda siempre abierta $P(V_3 = abierta) = 3 * 10^{-3}$
  - Se queda siempre cerrada $P(V_3 = cerrada) = 4 * 10^{-3}$

Tras esta introducción objetivo del sistema solamente queda definir la estructura de la red bayesiana dinámica que recoge la problemática de la distribución del fluido antes de realizar la simulación temporal.

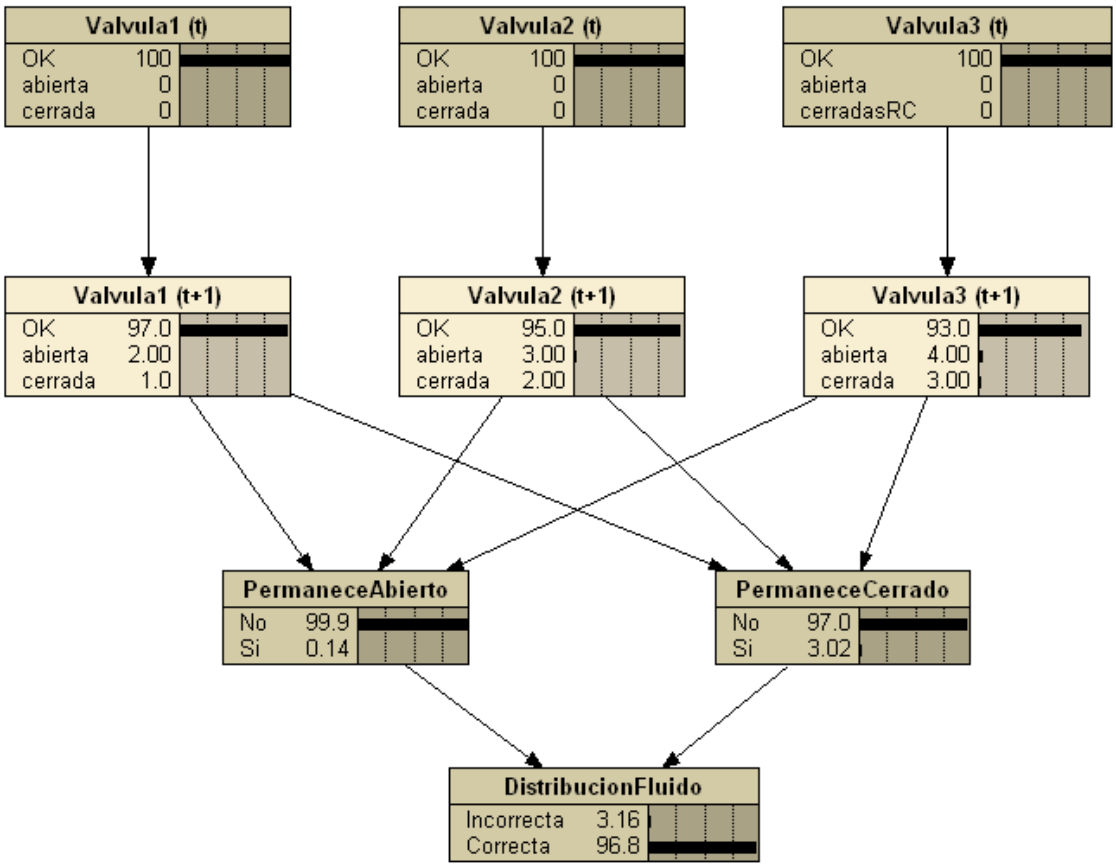

Imagen 6.5. RBD. Caso nº2.

En la red anterior las variables Valvula1 (t), Valvula2 (t) y Valvula3 (t) representan el estado de las válvulas en el instante t. Las variables Valvula1 (t+1), Valvula2 (t+1) y Valvula3 (t+1) representa el estado de las válvulas en el instante t+1. Por su lado PermaneceAbierto y Permanece Cerrado son variables que se utilizan para clasificar el fracaso del sistema, es decir, cuando el fluido está pasando constantemente o cuando está siendo retenido. En cuyo caso la distribución permanecerá abierta si Valvula1 (t+1) == "abierta" & (Valvula2 (t+1) == "abierta"|Valvula3 (t+1) = "abierta") y permanecerá cerrada si Valvula1 (t+1) == "cerrada" |(Valvula2 (t+1) == "cerrada" & Valvula3 (t+1) = "cerrada"). Y por último la variable DistribucionFluido determina si el sistema es controlable o no.

Como se puede observar en las variables Valvula1 (t+1), Valvula2 (t+1) y Valvula3 (t+1), las tasas de error han sido incorporadas en sus tablas de probabilidad condicional para representar esa probabilidad de error existente en el que si una válvula funciona correctamente en el momento t posteriormente en el instante t+1 deje de hacerlo.
Por consiguiente la simulación temporal tendrá como objetivo saber si la distribución del fluido es controlable. Este control vendrá representado por las probabilidades de los dos estados de la variable DistribucionFluido que nos mostrará el deterioro del sistema en el tiempo.

La imagen 6.6. muestra el resultado de la simulación temporal de 100 cortes de tiempo en la cual el objetivo del estudio consiste en observar las variables del modelo a lo largo del tiempo sin la introducción de ninguna evidencia simulando la ejecución a lo largo del tiempo del sistema de fluidos.

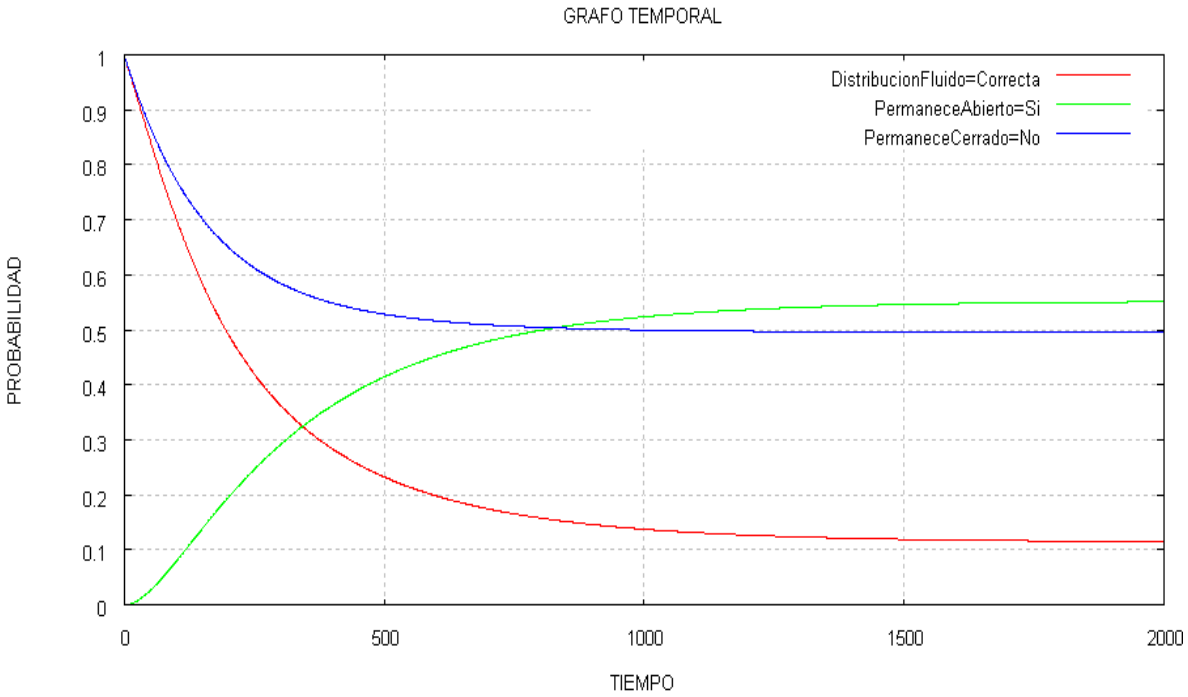

Imagen 6.6. Grafo temporal. Caso nº2.

La curva roja representa la probabilidad de que el sistema esté en funcionamiento en el tiempo t; las otras dos curvas representan la evolución de las probabilidades de los dos tipos de fracaso, la línea de color verde representa que el fluido está corriendo libremente por el sistema y la de color azul que el fluido permanece cerrado. Como se puede observar, la evolución de la probabilidad del funcionamiento correcto del sistema disminuye, este es debido al aumento de las tasas de errores de las válvulas con el paso del tiempo. La pendiente de dicha recta vendrá definida por el valor numérico asociado a las probabilidades de los errores aumentando está proporcionalmente a su valor. Como era de esperar el aumento de un tipo de error hace que disminuya el otro por eso en el caso de los dos tipos de fracasos vemos como los valores a lo largo del tiempo son complementarios.

Como ya se ha comentado, el ejemplo anterior es caso especial en el cuál no se produce ninguna evidencia a lo largo del tiempo. Esto no es del todo correcto en una situación real, ya que las piezas de los distintos sistemas que implementa algún sistema experto, ya sean válvulas u otros objetos, se cambian cuando la vida útil de estos ha expirado o se observa que su funcionamiento es incorrecto debido al deterioro sufrido.

Para poder recoger este hecho se ha modificado la red bayesiana dinámica inicial incorporando 3 nuevas variables que indicarán si las válvulas han sido cambias, arregladas o sustituidas por unas nuevas, por lo que las probabilidades de los errores se restauran al valor inicial como si el tiempo no hubiese afectado al funcionamiento

de las mismas. La imagen 6.7. recoge la red bayesiana [5] con estas nuevas modificaciones.

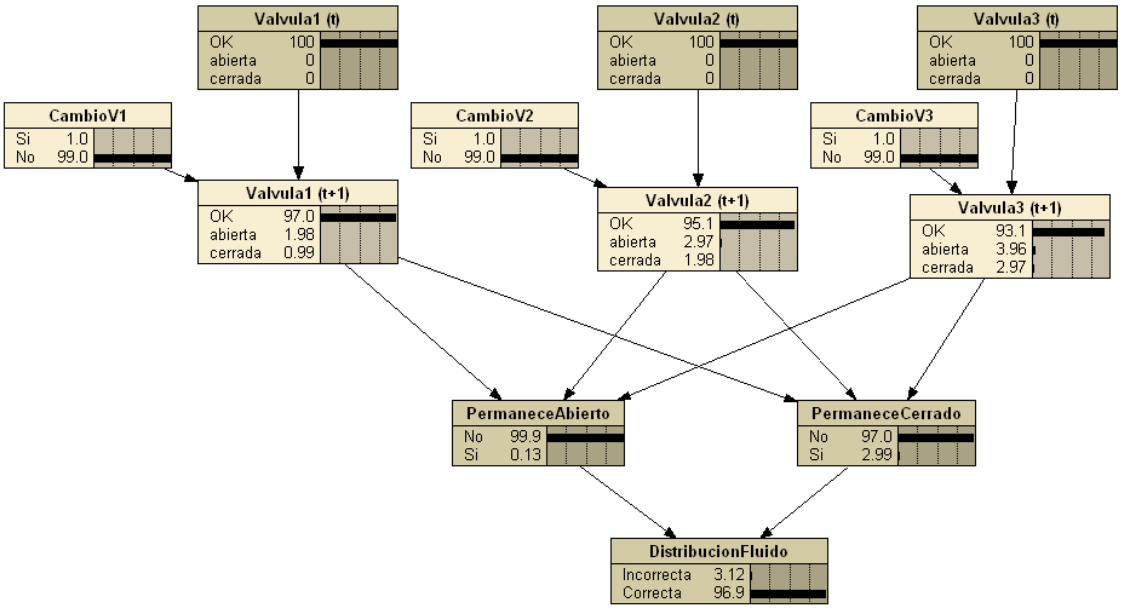

Imagen 6.7. RBD. Caso nº2 modificado.

Si con esta nueva configuración realizamos la simulación temporal, evidenciando un cambio de válvulas cada cierto periodo de tiempo, observamos como el sistema de fluidos recupera su funcionamiento inicial y empieza a decaer según pasa el tiempo hasta que se produzca un nuevo cambio de válvulas. El gráfico asociado a esta cuestión planteada se recoge en la imagen 6.8. donde se observa que cuando se produce un cambio en alguna de las válvulas la probabilidad de que el sistema funcione correctamente aumenta.

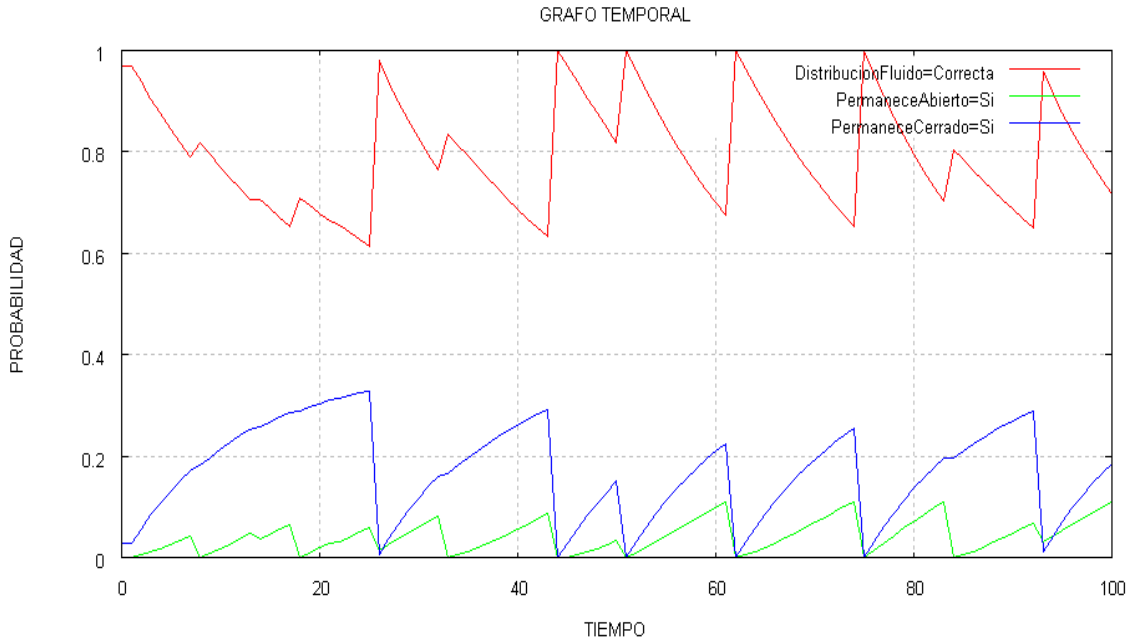

Imagen 6.8. Grafo temporal. Caso nº2 modificado.

[5] Nótese que se ha aumentado la tasa de error para realizar una simulación más corta de 100 cortes de tiempo y aumentar la pendiente de degradación de las válvulas.

6.3. *Tercer caso práctico*

Para este último caso, se expondrá un ejemplo de seguimiento y control de caídas de una persona presentado en (31) y (55). El enfoque se centra en la evaluación de una situación real de un individuo sobre la base de un nivel 2 de fusión de procesamiento de la información, recibida a través de la representación causal de una red bayesiana. En este ejemplo, se considera que habrá una serie de sensores y de componentes capaces de captar las características de los nodos de observación que serán los encargados de evidenciar las variables que nos permitan evaluar las situaciones específicas de alto nivel. Así este sistema de seguimiento y control de caídas proporcionará una visión real del estado del individuo a lo largo del tiempo.

La entrada de nuestra interfaz de simulación de redes bayesianas dinámicas vendrá proporcionada por la matriz de notificación instantánea, que a través de sensores que contendrán el individuo de seguimiento, proporcionará las evidencias de las variables de observación. La figura 6.2. representa todo el proceso de captación, procesamiento, representación y estimación del proceso de seguimiento y control de caída que daría pie a la creación de un sistema experto asociado a dicha problemática. En nuestro caso nos centraremos en la representación visual y simulación de la red bayesiana dinámica pues todo el tratamiento, procesamiento de los datos y deducciones de las variables está recogida en los documentos anteriormente mencionados.
La capa de simulación representa las acciones realizadas por el humano en tiempo real durante un proceso normal de actividad física, como son caminar o correr. La capa de sensores será la encargada de trasmitir las acciones del humano a través de observaciones realizadas por distintos dispositivos, los cuales, pueden tener un ruido asociado o cometer un cierto error de precisión. Por último la capa de estimación estaría compuesta por dos partes. La primera, constaría del procesamiento en bruto de los datos recibidos, donde se realizarían las tareas necesarias hasta acondicionar los datos a las variables de la red bayesiana. Esta red bayesiana dinámica sería la segunda parte, que estaría formada por la representación visual y simulación asociada al problema.

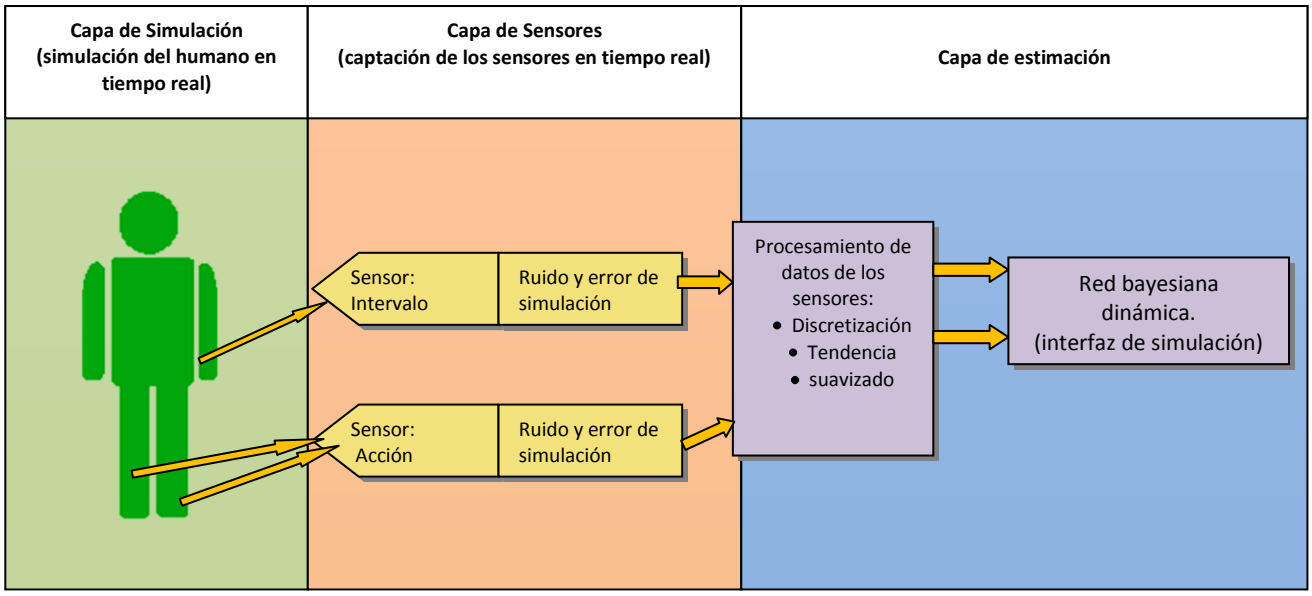

Figura 6.2. Visión global de caso práctico nº3.

La imagen 6.9. presenta la red bayesiana asociada al problema de control de seguimiento y advertencia de caída de una persona al caminar.

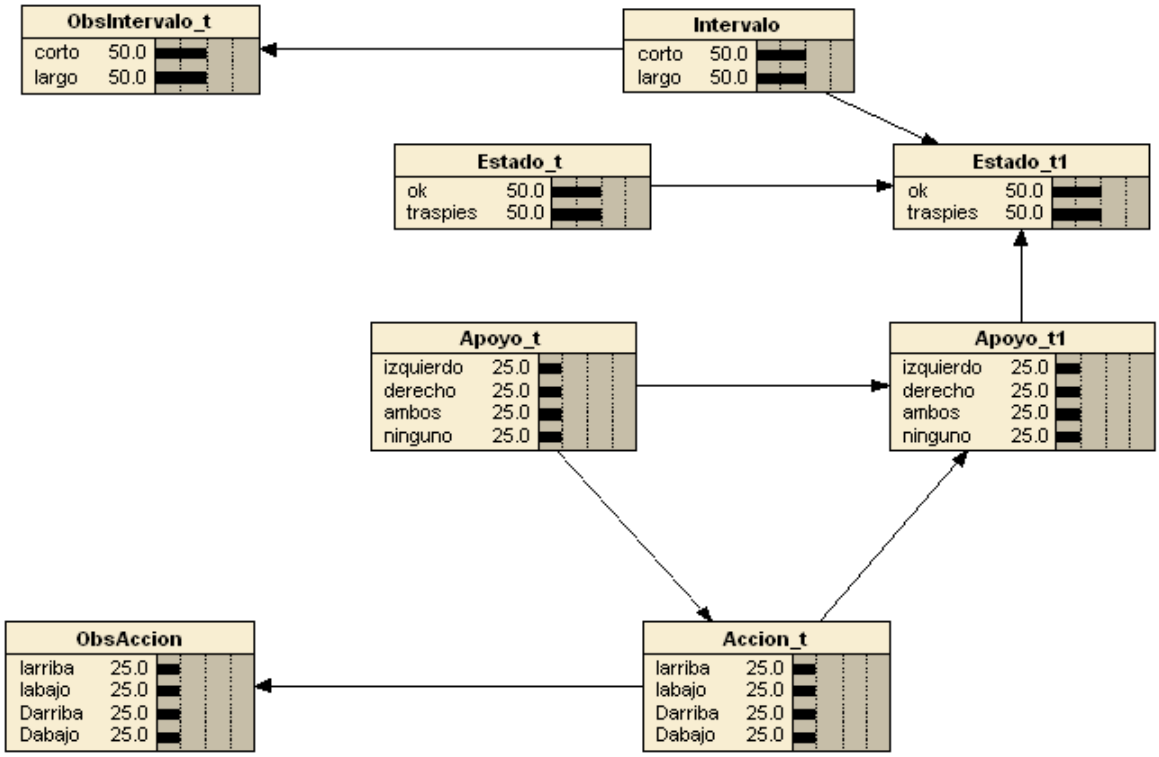

Imagen 6.9. RBD. Caso práctico nº 3.

Cuando suponemos que un individuo camina normalmente, éste suele ir alternando entre los pies izquierdo y derecho. Las tablas de probabilidad condicional se han recogido estudiando los casos normales tras el proceso de caminar de una persona y suponiendo que el proceso es simétrico tanto para el pie izquierdo como para el pie derecho. Aun así, el problema de detección de caída y tiene limitaciones, por ejemplo, no se manejan los casos en que ambos sensores de pie proporcionen datos al mismo tiempo o el individuo se siente.

Si hay cualquier variación en los patrones de marcha para un paciente determinado, por ejemplo, si tiene aluna pierna lesiona o alguna minusvalía, la red bayesiana dinámica se puede personalizar, variando los parámetros de probabilidad o removiendo la hipótesis de que el pie izquierdo y el derecho son exactamente simétricos.

La descripción de las variables asociadas a la red anterior son:

- El nodo “Apoyo_t”, representa la sujeción de una persona asociada a la acción de caminar, la cual en un momento determinado, puede tener solamente como pie de apoyo el izquierdo, el derecho, ambos o ninguno (en cuyo caso se ha caido).
- El nodo “Apoyo_t1”, representa la sujeción que tendrá la persona en el instante de tiempo t+1. Esta variable mostrará la probabilidad del siguiente apoyo siguiendo el proceso normal de caminar de una persona.

- El nodo “Accion_t”, recoge simboliza la variable de acción tras un evento realizado por la persona pudiendo haber levantado o bajado cada uno de los pies. Sus posibles estados son Iarriba (pie izquierdo subiendo), Iabajo (pie izquierdo subiendo), Darriba (pie derecho subiendo) y Dabajo (pie derecho bajando).
- El nodo “ObsAccion”, es la variable de observación de la acción asociada al individuo la cuál vendrá determinada por el sensor encargado de recoger dicho evento.
- El nodo “Estado_t”, representa la probabilidad de que en el momento actual el individuo tenga un tropiezo o no. El estado “ok” representa el estado normal en el que no ha habido una caída.
- El nodo “Estado_t1”, representa la probabilidad de que en el siguiente corte de tiempo el individuo tenga un tropiezo.
- El nodo “Intervalo”, representa el intervalo de tiempo entre los pasos del individuo. Este tiempo se ha considerado como “corto” para intervalos menores de 5 segundo (tiempo considerado para un patrón normal entre pasos) y “largo” para valores superiores. Siendo esta discretización modificable para adaptarse a diferentes individuos.
- El nodo “ObsIntervalo”, recoge las evidencias ofrecida por el sensor de intervalos que nos indicará si el tiempo ha sido mayor o menor a valor límite considerado entre dos pasos de tiempo.

A continuación, se verá el comportamiento de este modelo de red bayesiana dinámica[6] con el conjunto de parámetros asociados a un individuo en un proceso normal de paseo (la secuencia de observaciones será Iarriba, Iabajo, Darriba y Dabajo). Las observaciones entorno a los pies se irán alternando en orden correspondiente al proceso de caminar normal de un individuo, dicho ejemplo puede observarse en la imagen 6.10. donde se ve claramente que la secuencia una vez iniciado el proceso, a partir de t = 2 (que es cuando se estabiliza) la sucesión de pies de apoyo se va alternando entre el derecho, ambos e izquierdo.

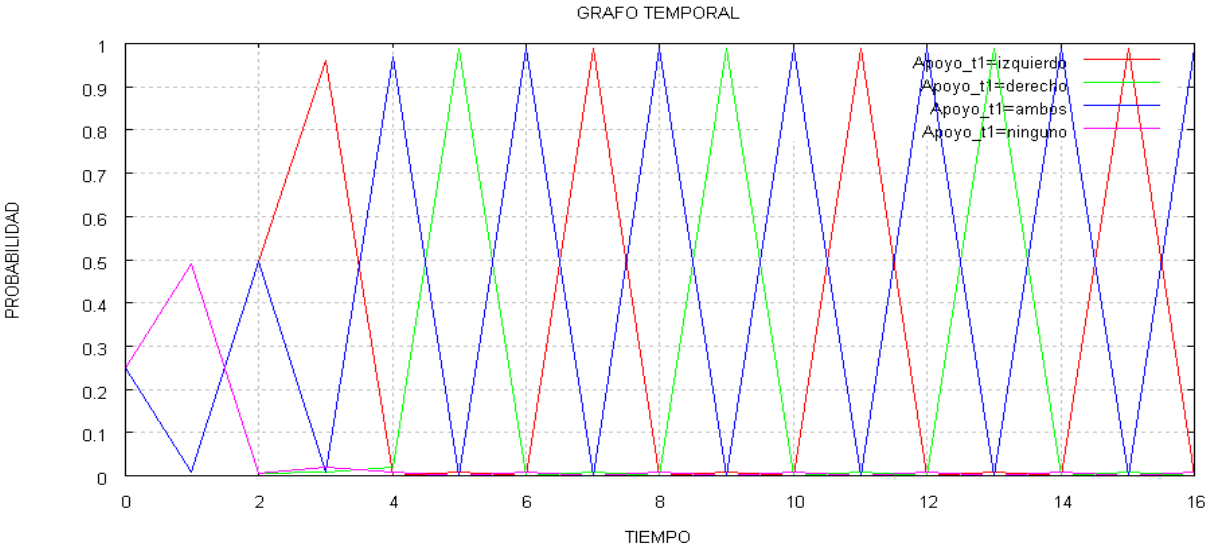

Imagen 6.10. Grafo temporal. Movimiento al caminar. Caso nº 3.

[6] Se ha llevado a cabo la simulación con las relaciones temporales entre los variable “Apoyo_t→Apoyo_t1” y “Estado_t→Estado_t1”

Como era de esperar si el proceso de caminar del individuo es estable, no ha habido ningún obstáculo y por tanto ni ninguna variación en la sucesión de apoyos, su estado de equilibrio será correcto teniendo una pequeña probabilidad de sufrir algún traspiés. La figura 6.11. muestra el grafo temporal asociado al caso de observaciones anterior.

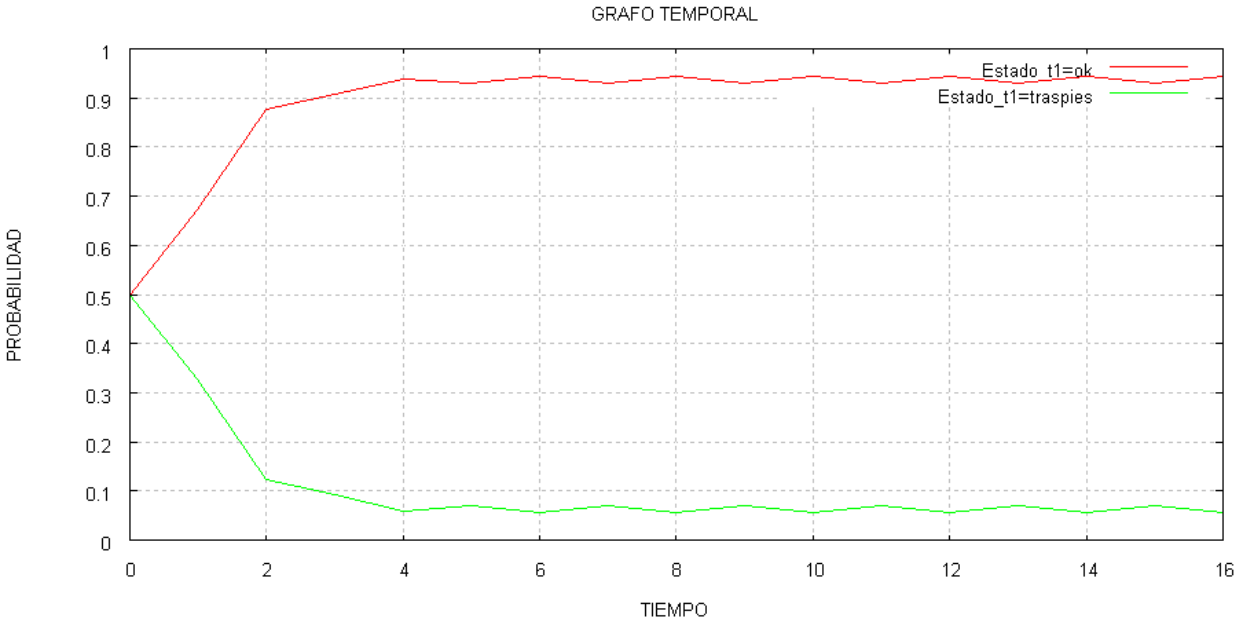

Imagen 6.11. Grafo temporal del estado del individuo al caminar. Caso nº3.

En el proceso normal de caminar del individuo anterior, no se ha dado la situación en el que por alguna casualidad la persona se encuentre con algún obstáculo o dificultad. Este hecho ocasionaría que el sensor de observación de acción empezase a producir evidencias fuera de la secuencia común de pasos. Este caso puede verse reflejado en la imagen 6.12. donde a partir del tiempo t = 16, se observa que las probabilidades del pie de apoyo para la variable "Apoyo_t1" empieza a salirse de la secuencia y a ser impredecible. Como último apoyo evidenciado por la variable "ObsAccion" fue "Iabajo" en t = 19, las probabilidades de apoyo variarán entre ningún pie apoyado y el pie derecho.

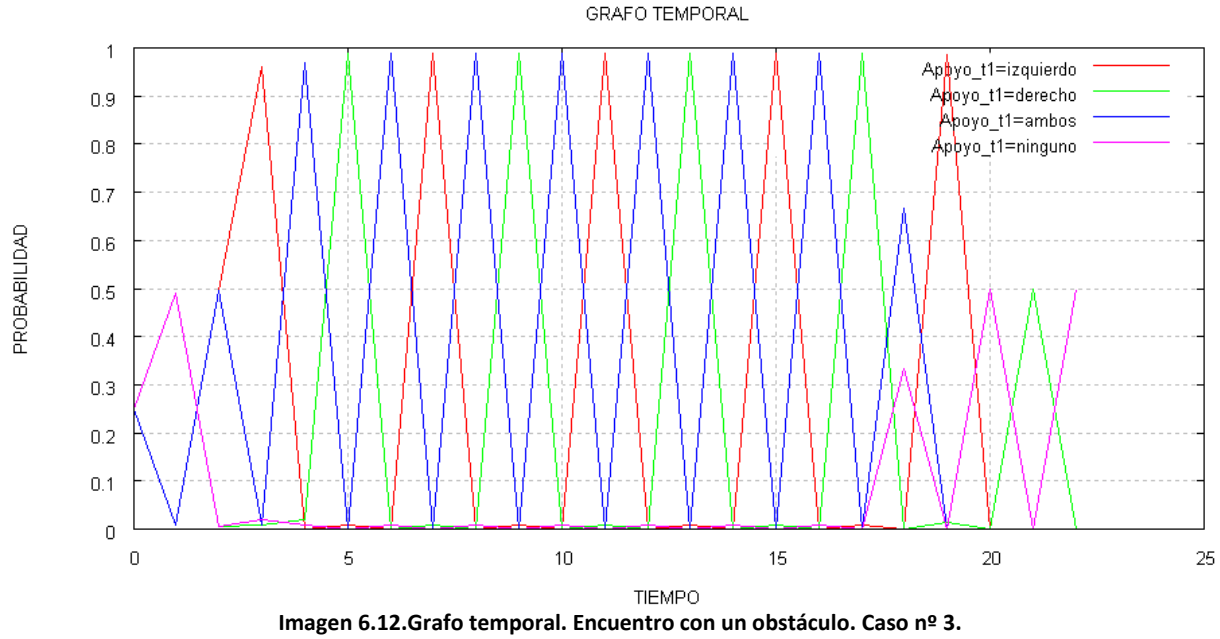

Imagen 6.12.Grafo temporal. Encuentro con un obstáculo. Caso nº 3.

Además de las observaciones del nodo “ObsAccion” en el caso anterior, se ha considerado que a partir de t = 17, el tiempo que han tardado de los sensores en emitir alguna observación ha sido largo (señales que tarden más de 5 segundos) en la variable “ObsIntervalo”. Este caso se observa en la imagen 6.13., donde aparte de las observaciones del nodo “ObsAccion”, la probabilidad de haber tenido un traspié crece al producirse este retardo en las observaciones de los pies del individuo, pudiendo haber sufrido una caída y no se haya recuperado todavía.

Imagen 6.13. Grafo temporal desviación al andar por un obstáculo. Caso nº 3.

Como se ha podido apreciar, este caso práctico de seguimiento y control de caída de un individuo contiene una gran cantidad de información y datos la cual tiene que haber sido precedida por estudios previos sobre el conocimiento del dominio, que bien puede haber contado con la ayuda de expertos. Y así, poder realizar el proceso de construcción de este sistema experto.
El refinamiento y acondicionamiento de las probabilidades y de las variables del caso son una parte importante que ocupará gran cantidad del tiempo del estudio en la que se tendrán que emplear distintas técnicas de análisis de datos que ayuden a entender y a optimizar el problema.

También, se ha podido comprobar cuál sería el procedimiento de transformación de un nivel 2 de fusión hasta la evaluación de una situación real de un individuo, donde a través de un procesamiento de la información recibida y con la ayuda de la representación causal de una red bayesiana, se ha construido un modelo complejo que nos predice dinámicamente el estado del individuo a través de las observaciones apreciadas por sensores.

A continuación, se muestra un caso similar basado en el anterior, sobre el seguimiento y control de caída, que ha sido modificado para poder representar el estado en el que se encuentra el individuo a través de las observaciones proporcionadas por son sensores, que nos permitirán evaluar la situación en la que se encuentra, es decir si el individuo está; parado, andando, corriendo, a punto de tener un traspiés o por el contrario se encuentra caído en el suelo.

Este modelo amplia el conocimiento sobre el modelo de control de caída inicial, mejorando la predicción y el reconocimiento de la situación. La nueva red bayesiana asociada al problema es:

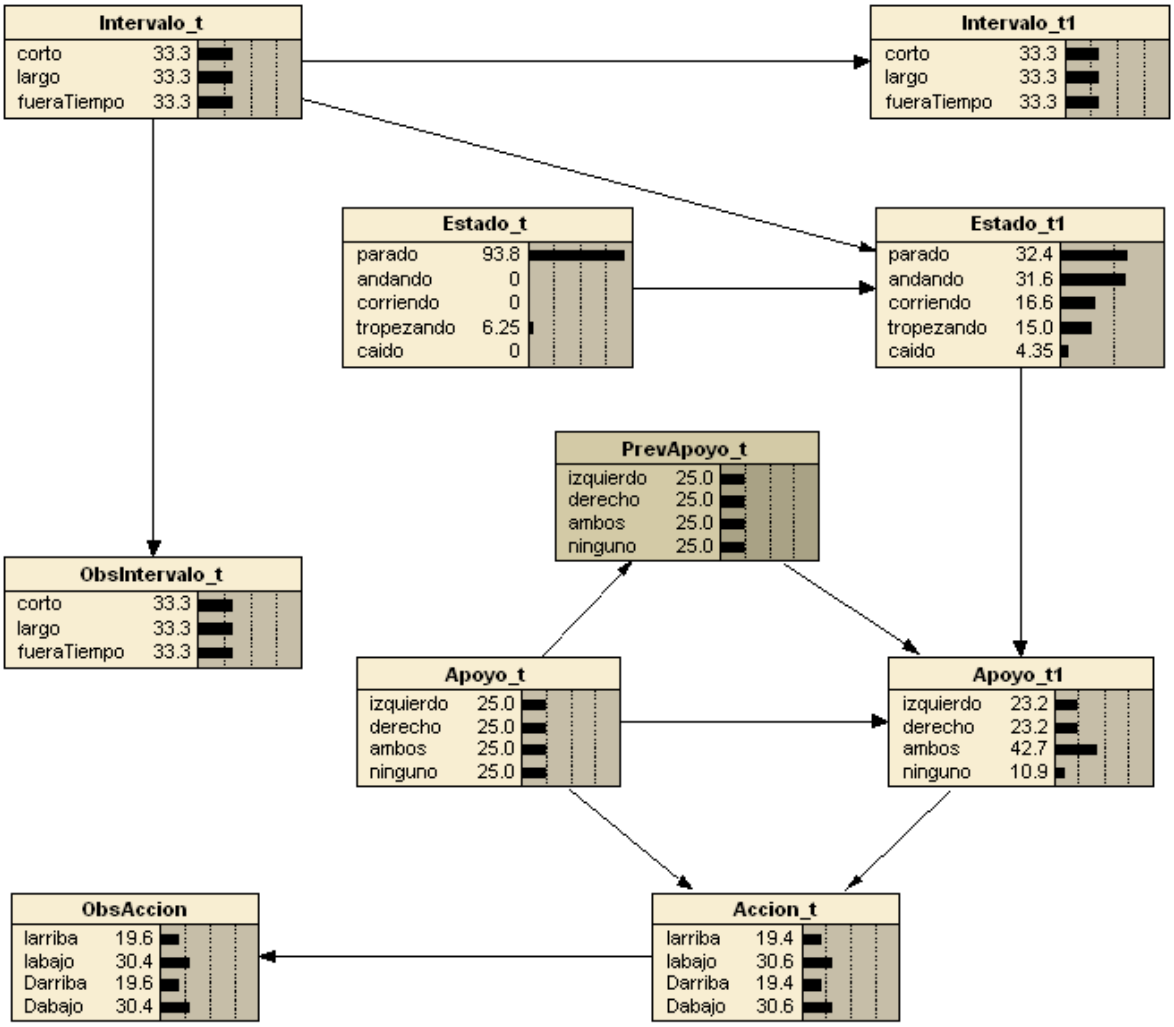

Imagen 6.14. RBD. Ampliación del caso nº 3.

Como se muestra en la imagen 6.14, las únicas diferencias estructurales respecto al caso inicial son las incorporaciones de los nodos “Intervalo_t1” y PrevApoyo_t, además de introducir un mayor número de estados en las variables “Intervalo_t”, “ObsIntervalor_t”, “Estado_t” y “Estado_t1”. Además se ha cambiado la relación de causalidad diregida del nodo “Estado_t1” al nodo “Apoyo_t1”.Este vínculo refleja la relación causal que existe entre el estado actual de movimiento del individuo en la secuencia de apoyos.

Por otro lado, la incorporación de un doble estado de apoyo, la variable "PrevApoyo_t" actúa de cómo nodo de transmisión, provoca que se tenga una mejor predicción en cuanto al apoyo próximo.
La incorporación de los nuevos estados a la variable "Estado_t" y "Estado_t1" representan una mayor exactitud sobre el estado de movimiento del individuo aportando una mayor información sobre el tipo de operación que realiza. En el caso previo, cuando se tenía solamente dos estados en la variable (ok y traspiés), existía la problemática de que el individuo se encontrase actualmente parado, tomando una mayor evidencia el estado de traspiés. Este problema se soluciona gracias a la adición de estos nuevos estados y una nueva discretización de los valores de las variables relacionadas con el tiempo. Siendo los nuevos estados "corto" para observaciones captadas en un tiempo inferior a 5 segundos, "largo" observaciones entre 5 y 9 segundos y por último, el estado "fueraTiempo" para observaciones que tarde más de 9 segundos.

La imagen 6.15. muestra el proceso de simulación[7] realizado en un proceso normal de seguimiento un individuo, donde se van alternando las piernas constantemente, pero donde el tiempo entre observaciones varia. El primer periodo será de observaciones largas, el segundo de periodos cortos y por el último observaciones que se producen fuera de tiempo donde a partir de t = 15 no se reciben señales de los sensores de los pies.

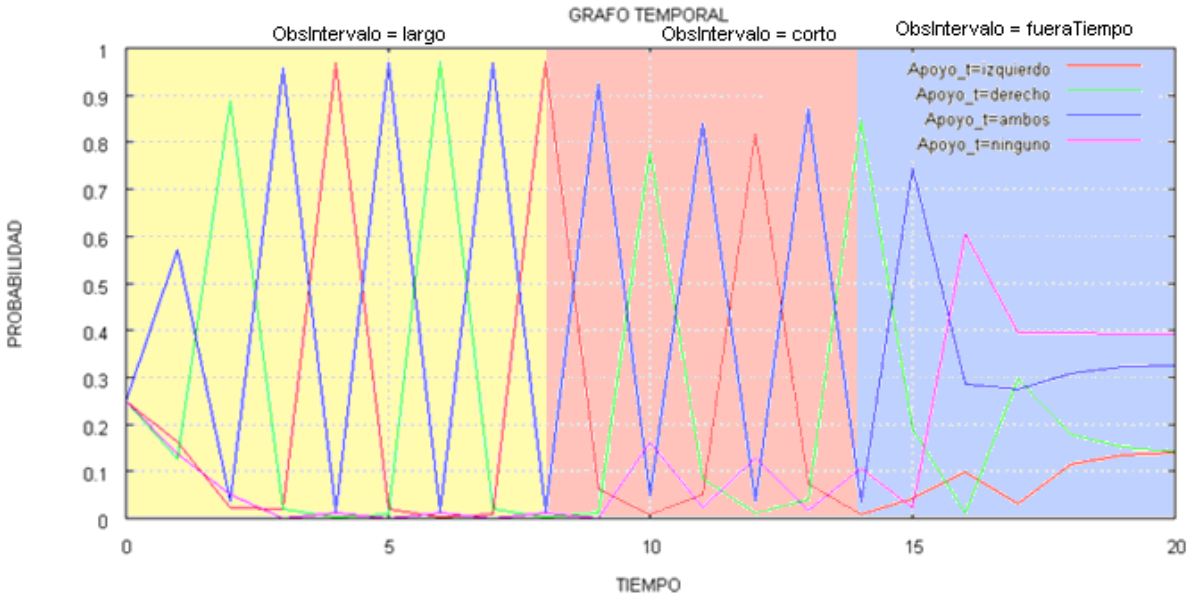

Imagen 6.15. Grafo temporal. Ampliación caso nº3.

En el grafo anterior vemos como efectivamente se va produciendo elcambio de pie de apoyo a lo largo de las observaciones pasando por la secuencia derecho, ambos, izquierdo, ambos, derecho…. A partir de t = 15 vemos como el proceso de observación de acciones del individuo a parado, ya que en un principio la creencia de que el "apoyo = ninguno" crece indicando que ha ocurrido algún suceso. Posteiormente al no recibir observaciones del sensor de accion de los piés los valores de apoyo empiezan a establecerle al valor normal de inicio.

[7] En este proceso de simulación se han considerado como relaciones temporales "Estado_t → Estado_t1", "Intervalo_t →Intervalo_t1" y "Apoyo_t → Apoyo_t1".

El gráfico temporal del estado del individuo asociado a las acciones y al estado de los pies de apoyo del caso anterior, se observa en la imagen 6.16. En el grafo se observa claramente, que para observaciones de un intervalo de tiempo largo la creencia de que el individudo está andado crece debido a que dicha actividad se realiza pausadamete.
Si por el contrario se mantiene las observaciones de las acciones constantes pero se varia el intervalo de tiempo entre observaciones, se observa cláramente que el estado individuo cambia, pasando por la creencia de que está actualmente corriendo (t = 9) y por tanto aumentando la probabilidad de que tropiece.
En el último tramo de estudio entre t [15, 20], donde se dan observaciones fuera de tiempo y donde detienen las evidencias de acción, se observa claramente que en t =15 al parár las observaciones, la evidencia de que el individuo se ha cadio es mayor debido a que la creencia de que se tropezase en el instante anterior era la que mayor probabilidad tenía. Posteriormente pata t > 15 se ve claramente que la incertidumbre varía entre dos posibles estados al no recibir observaciones de acción, y son que el individuo se haya parado o que siga en el suelo caido.

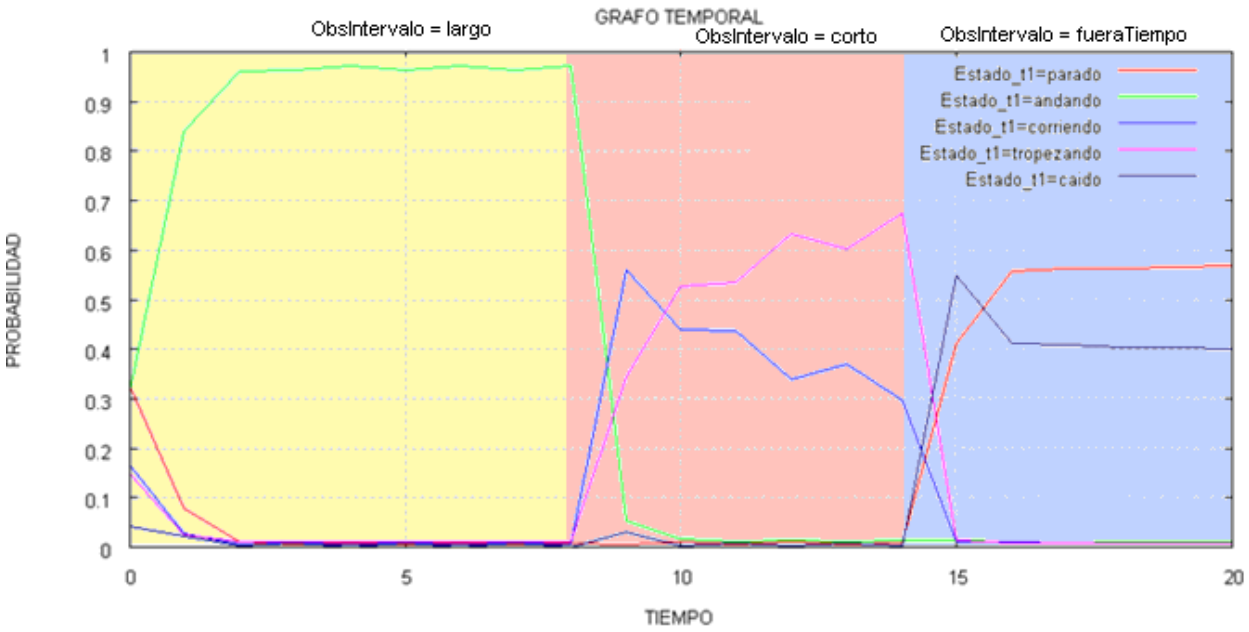

Imagen 6.16. Grafo temporal del estado del individuo. Ampliación caso nº 3.

Capítulo VII. Conclusiones

Tras la finalización del desarrollo de este proyecto los resultados obtenidos pueden calificarse de muy satisfactorios, ya que se han conseguido todos los objetivos que se propusieron.

Por un lado se han recogido una serie de características de los sistemas expertos que nos han permitido entender los distintos tipos que existen y la manera de poder representarlos. Posteriormente se ha podido presentar más concretamente, una de las técnicas que está en plena expansión de investigación como es el empleo de las redes bayesianas en la resolución de sistemas expertos probabilísticos.
Ha sido en este apartado de las redes bayesianas donde se ha dado un mayor énfasis en cuando a la documentación aportada, ya que como se presentó en la introducción del proyecto, uno de los objetivos es poder ilustrar las herramientas y directrices necesarias para futuros trabajos, que permitan ser un punto de referencia a posibles estudiantes que desarrollasen algún sistema experto dentro del ámbito de las redes bayesianas.

También ha sido importante, que durante la realización de este trabajo se han probado distintas herramientas para la utilización de redes bayesianas, como han sido Hugin Lite, BNJ, InferNet, BNet, Elvira, Bayesian, jBNC y Netica. De los cuales solamente algunos recogían ciertas de las características deseable en nuestro proyecto. A parte de ello, en el momento de analizar las herramientas a utilizar, se ha considerado la disponibilidad de éstas, se han evaluado las librerías, los lenguajes de programación, los costes de licencia, la facilidad de uso, etc. El software Netica, finalmente elegido, se ajustaba a las necesidades del proyecto además de haber sido utilizado con anterioridad, por lo que se consideró como la opción más viable. A pesar de ser este software un producto comercial, la versión de prueba (limitada en cuando al número nodos en la red) nos ha permitido explorar las características del producto y emplear la librería que dispone para la realización del sistema final que nos ha permitido trabajar con la aplicación Netica.

El desarrollo de la interfaz ha sido uno de los procesos más importantes ya que su realización aporta la herramienta necesaria para poder tratar problemas de razonamiento probabilístico en el tiempo a través del empleo de de las redes bayesiana dinámicas. La comunicación entre la aplicación Netica y la interfaz hacen que el sistema implementado, sirva de herramienta eficaz en el tratamiento temporal haciendo que la visualización de la red sea constante y dando soporte a la simulación temporal no ofrecida por la primera.

En cuanto a los dos modos ejecución realizados, el manual y el automático, ofrecen dos comportamientos perfectamente diferenciados. El primero ofrece las características necesarias para la exploración y observación de casos prácticos en los cuales la incorporación de los hallazgos vienen aportados por el usuario, haciendo que las probabilidades evolucionen a lo largo del tiempo por la introducción de estas evidencias.

En cuanto al funcionamiento del segundo, su implementación va más allá de la creación de una herramienta para el estudio de problemas probabilísticos en el tiempo a través de redes bayesianas, sino que permite su utilización en paralelo con otros sistemas de análisis permitiendo representar el conocimiento para el razonamiento en condiciones de incertidumbre.
Las salidas de los sistemas análisis (se considera como salida, la evidencia de haberse producido un hallazgo en una variable) serán la entrada en nuestra interfaz, consiguiendo por un lado la representación visual en tiempo real del estado del conocimiento incierto y por otro la evolución del sistema a lo largo del tiempo.

En cuanto al tamaño de los problemas para la resolución a través de las redes bayesianas, se ha visto que en la actualidad, existen algoritmos exactos que permiten el cálculo de las probabilidades en redes de varios cientos de nodos, siempre que no haya demasiados bucles, que para modelos más complejos, hacen que las redes bayesianas puedan resultar ineficientes salvo que se empleen métodos de aproximación adaptados a la naturaleza del problema.

Por otro lado, con la incorporación de casos prácticos reales, se ha conseguido enfocar el desarrollo de distintos sistemas expertos para la evaluación de situaciones (nivel 2 de fusión) los cuales, nos han permitido emplear la metodología de las redes bayesianas para su resolución. Además, estos casos han servido de pruebas prácticas para el análisis de la interfaz desarrollada, sirviendo de modelos para la comprobación del funcionamiento, de su empleo y de su utilidad.

Por último, en cuanto a líneas de trabajo a seguir en un futuro, a partir de este proyecto, probablemente pasen por dos aspectos. El primero, hace referencia a la construcción de nuevas funcionalidades en la interfaz implementada, que permitan o incorporen funciones de otros aspectos de las redes bayesianas. Y el segundo, como referencia de documentación para futuros proyectos, proporcionando un análisis sobre las características de las redes bayesiana dinámicas y del software relacionado, que permite el uso de una interfaz de simulación, necesaria para la construcción de sistemas de evaluación de situaciones que traten la problemática de la incertidumbre a lo largo del tiempo.

Capítulo VIII. Gestión del proyecto

A continuación se presentan los aspectos más importantes relacionados con la gestión del proyecto, desglosando la planificación realizada y estimando los costes asociados al mismo.

La organización formal del proyecto consta de dos personas: Jesús García Herrero, en el rol de director de proyecto, y Daniel García Gallego, como autor de este trabajo. Las supervisiones casi quincenales, realizadas por parte del director del proyecto, del trabajo realizado no se incluirán dentro de la estructura de la planificación por considerarlas no significativas en cómputo de tiempo en relación con el desarrollo total del proyecto. Tampoco se han considerado estas supervisiones como hitos, ya que no han sido finalizaciones de tareas o trabajos a alcanzar.

8.1. Planificación

La planificación del proyecto se ha elaborado teniendo en cuenta la necesidad de entrega del proyecto fin de carrera a finales del mes de febrero del 2010, considerando como fecha límite lunes 22.

El desarrollo del proyecto se ha dividido en varias fases para así poder controlar de una manera más clara los aspectos más relevantes del mismo. Además cada fase constará de subtareas que describirán brevemente el proceso llevado en cada una de ellas.

Las fases más importantes que se han considerado en el proyecto junto con una pequeña definición son:

- Documentación previa: tarea consistente en el proceso de búsqueda y adquisición de conocimiento sobre los distintos aspectos más relevantes de las temáticas del proyecto.
- Análisis: proceso de extracción de los requisitos de usuario propios del sistema a desarrollar.
- Diseño: tarea que consiste en la definición de la arquitectura del sistema y del entorno tecnológico que le va a dar soporte, junto con la especificación detallada de los componentes del sistema de información.
- Instalación del entorno de trabajo: realización de las configuraciones e instalaciones previas de los programas necesarios para la implementación.
- Implementación: plasmación del diseño realizado en el leguaje de programación empleado codificando los diagramas de clases esbozados.
- Pruebas: proceso consistente en comprobar que el software realiza correctamente las tareas indicadas en la especificación del problema.
- Memoria: redacción concerniente a la documentación propia del proyecto.

A continuación, se muestra la planificación de tiempo, empezando por las tareas que componen el proyecto así como las dependencias entre las mismas para posteriormente ser completado con el diagrama de Gantt correspondiente, el cual permite tener una visión más rápida y global de la planificación del proyecto.

8.1.1. Tareas

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	**Interfaz redes bayesianas dinamicas**	**125 días?**	**mar 01/09/09**	**lun 22/02/10**	
2	**Documentación previa**	**30 días**	**mar 01/09/09**	**lun 12/10/09**	
3	Sistema expertos	4 días	mar 01/09/09	vie 04/09/09	
4	Redes bayesianas	18 días	lun 07/09/09	mié 30/09/09	3
5	Herramientas y uso	8 días	jue 01/10/09	lun 12/10/09	4
6	**Análisis**	**13 días?**	**vie 02/10/09**	**mar 20/10/09**	
7	Captura de requisitos	4 días?	vie 02/10/09	mié 07/10/09	
8	Aprendizaje Netica	4 días	vie 09/10/09	mié 14/10/09	7
9	Estudio de API	7 días	lun 12/10/09	mar 20/10/09	
10	**Diseño**	**11 días**	**mié 21/10/09**	**mié 04/11/09**	**6;9**
11	Diseño de la interfaz	3 días	mié 21/10/09	vie 23/10/09	
12	Diseño conceptual	8 días	lun 26/10/09	mié 04/11/09	11
13	**Instalación del entorno de trabajo**	**2 días?**	**jue 05/11/09**	**vie 06/11/09**	**10**
14	Instalación de Netica	1 día	jue 05/11/09	jue 05/11/09	
15	Instalación del entorno VS 2005	1 día?	jue 05/11/09	jue 05/11/09	
16	Instalación de la API	1 día?	vie 06/11/09	vie 06/11/09	15
17	**Implementación**	**30 días**	**lun 09/11/09**	**vie 18/12/09**	**13**
18	Implementación de la interfaz	5 días	lun 09/11/09	vie 13/11/09	
19	Implementación del modo manual	14 días	lun 16/11/09	jue 03/12/09	18
20	Implementación del modo automático	11 días	vie 04/12/09	vie 18/12/09	19
21	**Pruebas**	**10 días**	**lun 21/12/09**	**vie 01/01/10**	**17**
22	Modo manual	5 días	lun 21/12/09	vie 25/12/09	
23	Modo automático	5 días	lun 28/12/09	vie 01/01/10	22
24	Corrección de errores	10 días	lun 21/12/09	vie 01/01/10	
25	**Implantación**	**7 días?**	**lun 04/01/10**	**mar 12/01/10**	**21;24**
26	Manual de usuario Netica	4 días	lun 04/01/10	jue 07/01/10	
27	Manual de usuario del sistema	2 días	vie 08/01/10	lun 11/01/10	26
28	Instalación del sistema	1 día?	mar 12/01/10	mar 12/01/10	27
29	**Memoria**	**95 días?**	**mar 13/10/09**	**lun 22/02/10**	
30	Primera parte	27,5 días?	mar 13/10/09	jue 19/11/09	2
31	Segunda parte	30 días?	mar 12/01/10	lun 22/02/10	17;30

Figura 8.1. Tareas del proyecto.

8.1.2. Diagrama de planificación

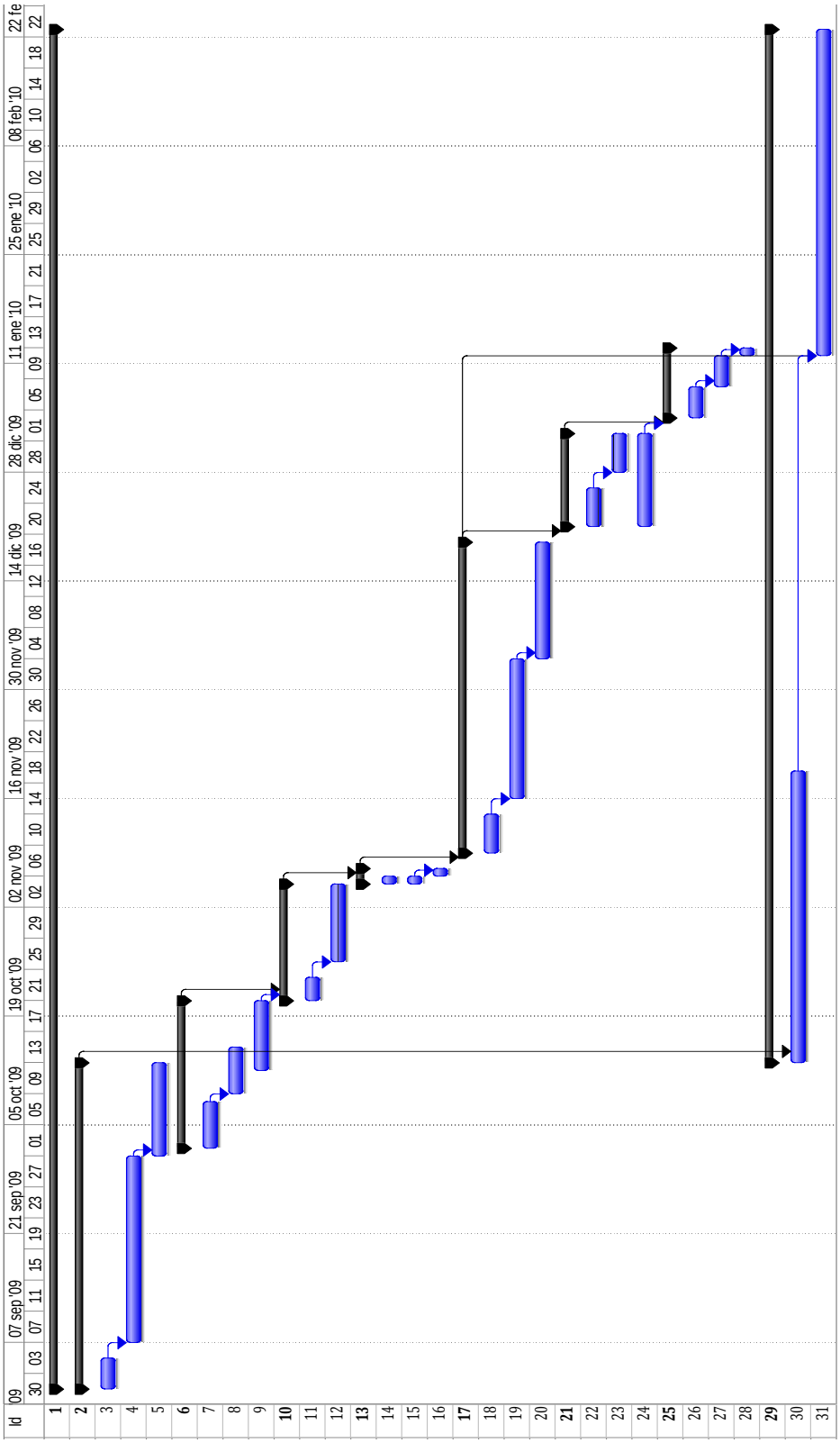

Figura 8.2. Diagrama de planificación.

8.1.3. Presupuesto

Para la elaboración del presupuesto se realizará la evaluación de los recursos humanos y de los recursos materiales (tanto hardware como software) necesarios para la realización del proyecto. En el coste de los recursos humanos se realizará el cálculo basándose en la planificación expuesta en el apartado anterior, y en una estimación de los posibles integrantes del grupo del trabajo que formasen parte del proyecto. Esta estimación será relevante para el coste de los recursos humanos pues dependiendo de los roles necesarios en la gestión del proyecto se tendrán unos costes distintos en el sueldo de cada individuo. En nuestro caso, como ya se comentó en la introducción del apartado 7. Gestión del proyecto, los únicos roles necesarios para el cálculo de costes de los recursos humanos serán el de jefe de proyecto y el de un ingeniero júnior.

8.1.3.1. Recursos software

Los productos software que se han empleado ya fueron citado en el apartado 2.2. Entorno de trabajo, donde se resume el porqué de su utilización. En dicho apartado faltan ciertos programas, que a continuación se recogerán para no englobarlos en la suite ofimática de Microsoft® Office, y tener un mayor nivel de detalle en el desglose de gastos ocasionados por los recursos software empleados. El listado de programas completos es:

- Microsoft® Windows XP
- Microsoft® Visual Studio 2005
- Microsoft® Word 2003
- Microsoft® Project 2003
- Microsoft® Visio 2003
- Aplicación Netica
- Gnuplot
- API Netica

El coste de las licencias necesarias para el desarrollo del proyecto se muestra en la tabla 8.1., en ella se aprecia el coste mensual de cada software considerando como tiempo de amortización el periodo de 12 meses. En nuestro caso no será necesaria dicha estimación gracias a las licencias gratuitas que ofrece el servicio MSDN Academy.

Tabla 8.1. Recursos software.

CONCEPTO	LICENCIA	COSTE MENSUAL (€/mes)	MESES	TOTAL
Microsoft® Windows XP	MSDN Academy	0	0	0
Microsoft® Visual Studio 2005	MSDN Academy	0	0	0
Microsoft® Word 2003	MSDN Academy	0	0	0
Microsoft® Project 2003	MSDN Academy	0	0	0
Microsoft® Visio 2003	MSDN Academy	0	0	0
Aplicación Netica	Versión Evaluación	0	0	0
Gnuplot	Licencia GNU	0	0	0
API Netica	Versión Evaluación	0	0	0
		TOTAL RECURSOS SOFTWARE		**0€**

8.1.3.2. Recursos hardware

La capacidad de los recursos hardware tienen que ser lo suficientemente potentes para que los resultados que se esperen del proyecto no se vean afectados en pérdidas de rendimiento y por tanto en tiempo. El equipo que se describirá en esta sección no tiene porque ser el equipo mínimo orientativo requerido por las aplicaciones software, pero al desarrollarse todo el proyecto en él, se considera necesario presentar sus características. El conjunto de recursos hardware empleados en el proyecto son:

- Ordenador portátil con las siguientes características:
  - Intel® Core™ 2 Duo CPU T5750 @ 2.00 GHz
  - Memoria RAM 2 GB DDR2 a 533 MHz
  - Tarjeta gráfica Mobile Intel® 965
  - Pantalla WXGA TFT 15,4"
  - Disco duro SATAII 120 GB
  - Unidad lectora/gradora DVD-RW
  - Wi-Fi™ Realtek Wireless Lan 802.11b/g
  - Tarjeta de red Realtek Fast Ethernet LAN 10/100

Tabla 8.2. Recursos hardware.

CONCEPTO	COSTE MENSUAL (€/mes)	MESES	TOTAL (€)
PC utilizado	700 €/12 meses	6	350,00 €
	TOTAL RECURSOS HARDWARE		350,00 €

8.1.3.3. Recursos humanos

Tras el estudio de la planificación inicial y la estimación de esfuerzo, se han calculado las horas de dedicación necesarias para los dos roles participantes en el proyecto. En el caso de ingeniero júnior se ha estimado que el esfuerzo ha sido de 4 horas diarias a los largo de la duración de todo el proyecto. Para el cómputo de coste del jefe de proyecto se ha considerando que ha habido una participación de 2 horas cada quince días.

Una vez estimado el tiempo de dedicación de cada uno de los roles solamente queda detallar el coste por cada hora de inversión en el proyecto. La tabla muestra el cálculo de costes de contratación y de recursos humanos.

Tabla 8.3. Recursos humanos.

CATEGORÍA	COSTE HORA (€)	DÍAS	HORAS	TOTAL (€)
Jefe de Proyecto	60 €	12	24	1.440,00 €
Ingeniero júnior	30 €	125	500	15.000,00 €
			TOTAL RECURSOS HUMANOS	**16.440,00 €**

8.1.3.4. Costes indirectos y bienes fungibles

Se ha considerado como costes indirectos todas aquellas facturas relativas a la contratación de servicios necesarios para la realización del proyecto. Estos pueden resumirse como las prestaciones imprescindibles de la luz (para el funcionamiento del equipo personal de trabajo así como otros dispositivos necesarios para la comunicación, la contratación del servicio de línea telefónica y ADSL y el consumo mensual originado por el uso del transporte hasta el lugar de trabajo.
El periodo de contratación de los costes asociados a la luz y la tarifa plana 3G, queda establecida desde el martes día 01 de septiembre del 2009 (inicio del proyecto) al jueves día 28 de febrero del 2010 (coincidiendo con la fecha del final del mes de la entrega el proyecto).

En cuanto al coste de los bienes fungibles se ha considerado como todo aquel material de oficina necesario así como los materiales necesarios para la impresión.
El material de oficina corresponde al conjunto de bolígrafos, rotuladores, folios, etc.
El material de impresión corresponde a los cartuchos de tinta necesarios para la impresión y encuadernación de los distintos documentos que se van a entregar la hora de presentar el proyecto.

Tabla 8.4. Costes indirectos y bienes fungibles

CONCEPTO	COSTE MENSUAL (€)	MESES	TOTAL (€)
Costes indirectos	98,29 €	6	589,74 €
Bienes fungibles	10,00 €	6	60,00 €
TOTAL COSTES INDIRECTOS Y BIENES FUNGIGLES			**649,74 €**

8.1.3.5. Coste total del proyecto

A continuación se muestra en la tabla 8.5., el resumen total de los costes originados por el proyecto.

Tabla 8.5. Coste total del proyecto.

CATEGORÍA	COSTE (€)
Recursos software	0,00 €
Recursos hardware	350,00 €
Recursos humanos	16.440,00 €
Costes inmuebles y bienes fungibles	649,74 €
TOTAL	**17.439,74 €**
I.V.A (16%)	**2.790,36 €**
COSTE TOTAL DEL PROYECTO	**20.230,10 €**

Por consiguiente el coste total del proyecto asciende a 20.230,10 €; veinte mil doscientas treinta euros con diez céntimos.

Anexo A: Software disponibles

Debido a la evolución sufrida en los últimos años, y más con el uso de redes bayesianas para el razonamiento de problemas con incertidumbre, ha ocasionado que aparezcan en el mercado una serie de paquetes y/o librerías que recogen las funcionalidades y las características de construcción, entrenamiento y evaluación de las redes bayesianas, dando por consiguiente, las herramientas necesarias para simplificar el proceso de programación.

En este anexo se quiere recoger, de manera resumida, dicho conjunto de utilidades para el desarrollo de redes bayesianas, permitiendo por tanto escoger aquella que mejor convenga al desarrollador o cliente para su proyecto.

El Software aquí presentado se basa en la lista de paquetes de software para redes bayesianas de Kevin Murphy (56) y en el apendice: Software Packages del libro "*Bayesian Artificial Intelligence*" (31). Por consiguiente, se intentará completar en la medida de lo posible la lista presentada sin adentrarse en características avanzadas de las herramientas (funcionamiento interno, representaciones, etc.), ni opciones de las interfaces gráficas de usuario.

El resumen del software presentado por Kevin Murphy se puede observar en las tablas A.1 y A.2. La primera de ellas recoge todo el software disponible actualmente en disposición y en la segunda tabla se presentarán las características de cada uno de ellos como son: la inclusión del código fuente, si disponen de una interfaz gráfica incluida, el lenguaje de programación en el que está desarrollado, etc.

Se ha decidido por comodidad particionar las tablas verticalmente debido a la extensión de los atributos característicos y así, poder representar más convenientemente la información.

En la Tabla A.1. Colección de software, se recogen los siguientes atributos:

- Nombre: seudónimo del paquete software o librería.
- Autor: referencia al programador, empresa comercial o universidad que ha desarrollado o desarrolla el programa.
- Dirección Web: lugar donde se puede encontrar una mayor información.

Tabla A.1. Colección de software

Nombre	Autor	Dirección Web
AgenaRisk	Agena	http://www.agenarisk.com/
Analytica	Lumina Decision Systems	http://www.lumina.com/
Banjo	Alexander J. Hartemink	http://www.cs.duke.edu/~amink/software/banjo/
Bassist	Universidad de Helsinki	http://www.cs.helsinki.fi/research/fdk/bassist/
BayesBuilder	Marcel Nijman, Ender Akay, and Wim Wiegerinck	http://www.snn.ru.nl/nijmegen/bayesbuilder.php3[8]
BayesiaLab	Bayesia	http://www.bayesia.com/en/index.php
Bayesware Discoverer	Bayesware	http://www.bayesware.com/
B-course	Universidad de Helsinki	http://b-course.cs.helsinki.fi/obc/
Belief net power constructor	Jie Cheng (Universidad de Alberta)	http://www.cs.ualberta.ca/~jcheng/bnpc.htm
BayesBlock	Universidad de Helsinki	http://www.cis.hut.fi/
Blaise	Keith Bonawitz, Vikash Mansinghka y Beau Cronin	http://publications.csail.mit.edu/abstracts/abstracts07/bonawitz/bonawitz.html
BNT	Kevin Murphy	http://people.cs.ubc.ca/~murphyk/Software/BNT/bnt.html
BNJ	Universidad del estado de Kansas	http://bnj.sourceforge.net/index-es.html
BNL	Frank Rijmen	http://www.mathworks.com/matlabcentral/fileexchange/13136
BUGS	Universidad de Medicina de St Mary's.	http://www.mrc-bsu.cam.ac.uk/bugs/
Causal Discoverer	Universidad de Vanderbilt	http://discover1.mc.vanderbilt.edu/discover/public/
CoCo+Xlisp	Badsberg (Universidad de Aalborg)	http://www.math.auc.dk/~jhb/CoCo/cocoinfo.html[9]
Clspace	Universidad Británica de Colombia	http://www.aispace.org/index.shtml
CRFtoolbox	Mark Schmidt y Kevin Murphy	http://people.cs.ubc.ca/~murphyk/Software/CRF/crf.html
DBNbox	Stephen Roberts	http://www.robots.ox.ac.uk/~parg/software.html

[8] Actualmente dicha página Web no se encuentra disponible, otra opción de consulta es http://www.snn.ru.nl/nijmegen/index.php?option=com_content&view=article&id=89&Itemid=212

[9] Actualmente dicha referencia no se encuentra disponible y no se ha hallado ninguna en sustitución.

Nombre	Autor	Dirección Web
Deal	Universidad de Aalborg	http://www.math.auc.dk/novo/deal[3]
Derivelt	Derivelt LLC	http://www.deriveit.com/modeling.htm
Elvira	Consorcio de universidades de España	http://www.ia.uned.es/~elvira/#proyecto
Ergo	Noetic systems	http://www.noeticsystems.com/ergo/index.shtml[10]
GDAGsim	Wilkinson (Universidad de Newcastle)	http://www.staff.ncl.ac.uk/d.j.wilkinson/software/gdagsim/
GeNle and SMILE	Decision Systems Laboratory, Universidad de Pittsburgh	http://genie.sis.pitt.edu/
GGM	West	http://xpress.isds.duke.edu:8080/softwarelinks/ggm.html[3]
GMRFSim	Håvard Rue	http://www.math.ntnu.no/~hrue/
GMTk	Jeff Blimes (Universidad de Washington)	http://ssli.ee.washington.edu/~bilmes/gmtk/
gR	Proyecto gR	http://www.ci.tuwien.ac.at/gR/
Grappa	Peter Green (Bristol)	http://www.stats.bris.ac.uk/~peter/Grappa/
HdBCS	Adrian Dobra, Quanli Wang, Liang Zhang y Mike West	http://www.stat.duke.edu/~adobra/hdbcs.html
Hugin Expert	Hugin	http://www.hugin.com/
Hydra	Warnes	http://sourceforge.net/projects/hydra-mcmc/
Infer.NET	John Winn, Tom Minka	http://research.microsoft.com/en-us/um/cambridge/projects/infernet/default.aspx
JAGS	Martyn Plummer	http://www-ice.iarc.fr/~martyn/software/jags/
Java Bayes	Fabio Gagliardi Cozman	http://www.pmr.poli.usp.br/ltd/Software/javabayes/Home/
LibB	Nir Friedman and Gal Elidan	http://www.cs.huji.ac.il/labs/compbio/LibB/
MIM	HyperGraph Software	http://www.hypergraph.dk/
Mocapy++	Universidad de Copenhagen	http://mocapy.sourceforge.net/
MSBNx	Microsoft	http://research.microsoft.com/en-us/um/redmond/groups/adapt/msbnx/
Netica	Norsys Software Corporation	http://www.norsys.com/
Bayes net Learner	Andrew Moore y Weng-Keen Wong	http://www.autonlab.org/autonweb/10530
PMT	Vladimir Pavlovic	http://www.cs.rutgers.edu/~vladimir/pmt/index.html
PNL	Intel	http://www.intel.com/technology/computing/pnl/[11]

[10] Actualmente dicha referencia no se encuentra disponible y no se ha hallado ninguna en sustitución.
[11] Actualmente dicha referencia no se encuentra disponible y no se ha hallado ninguna en sustitución.

Nombre	Autor	Dirección Web
Pulcinella	IRIDA(Universidad Libre de BruXelles)	http://iridia.ulb.ac.be/pulcinella/Welcome.html
RISO	Robert Dodier	http://riso.sourceforge.net/
Sam lam	AR Group California	http://reasoning.cs.ucla.edu/samiam/
LADR	Structure Data	http://www.structureddata01.com/
Tetrad	Carnegie Mellon University, Pittsburgh	http://www.phil.cmu.edu/projects/tetrad/tetrad4.html
UC Irvine	Dechter	http://graphmod.ics.uci.edu/group
UnBBayes	Mario Vieira	http://unbbayes.sourceforge.net/
Vides	John Winn	http://vibes.sourceforge.net/
WinMine	Microsoft	http://research.microsoft.com/en-us/um/people/dmax/WinMine/tooldoc.htm
XBAIES 2.0	Cowell	http://www.staff.city.ac.uk/~rgc/webpages/xbpage.html[4]

A continuación se explicarán las cabeceras de los atributos de la tabla A.2., que recoge el conjunto de características de los paquetes de software:

- Src = No: no se incluye el código fuente. En otro caso indica el leguaje de programación.
- API = No, significa que el programa no puede ser integrado en nuestro código propio. En otras palabras, que sólo puede ser ejecutado como módulo aislado.
- Exec = Ejecutable bajo W = Windows (95/98/NT/XP), U = Unix, M = Mac o C = cualquier máquina con un compilador.
- Cts = nudos continuos (latentes) soportados: G = (condicionalmente) nudos gaussianos soportados analíticamente, Cs = nudos continuos soportados por muestreo, Cd = nudos continuos soportados por discretización, Cx = nudos continuos soportados por algún método no especificado, D = nodos discretos soportados.
- GUI = si se incluye Interfaz Gráfica de Usuario.
- Params = si se permiten parámetros con aprendizaje.
- Struct = representa el tipo de estructura de aprendizaje, CI = significa que usa tests condicionales de independencia, No = no se disponen de estructuras y Si = si se permiten.
- Utility = indica si se permiten nudos de decisión.
- Free = 0 indica software gratuito (aunque posiblemente para uso académico). $ = software comercial (la mayoría tiene versiones gratuitas que están restringidas o reducidas de varios modos; por ej., el tamaña del modelo es limitado, los modelos no se pueden guardar o no hay API).
- Undir = representa el tipo de grafo soportado, U = solamente grafos no orientados, D = grafos orientados, UD = ambos tipos grafos (orientados y no orientados) y G = grafos de cadena (mezclados orientados/no orientados).
- Inference = que tipo de algoritmo de inferencia es usado, jTree = junction tree, varelim = variable (bucket) elimination, MH = Metropols Hastings, G = muestreo de Gibbs, IS = importance sampling, sampling = algún otro método de

MonteCarlos, polytree = algoritmo de PEARL restringido a un grafo sin ciclos, VMP paso de mensajes variacional, EP = propagación expectativa, none = no se soporta la inferencia (el programa es diseñado solamente para la estructura de aprendizaje de datos completamente observados).

Tabla A.2. Propiedades de los paquetes software[12]

Nombre	Src	API	Exec	Cts	GUI	Params	Struct	Utility	Free	Undir	Inference
AgenaRisk	No	Si	W,U	Cx	Si	Si	No	No	$	D	JTree
Analytica	Java	Si	W,U,M	Cd	No	No	Si	No	0	D	sampling
Banjo	Java	Si	W,U,M	Cd	No	No	Si	No	0	D	None
Bassist	C++	Si	U	G	No	Si	No	No	0	D	MH
BayesBuilder	No	No	W	D	Si	No	No	No	0	D	-
BayesiaLab	No	No	-	Cd	Si	Si	Si	No	$	CG	JTree, G
Bayesware Discoverer	No	No	W,U,M	Cd	Si	Si	Si	No	$	D	-
B-course	No	No	W,U,M	Cd	Si	Si	Si	No	0	D	-
Belief net power constructor	No	Si	W	D	Si	Si	CI	No	0	D	-
BayesBlock	Python/C++	Si	-	Si	No	Si	No	No	0	-	VMP
Blaise	Java	Si	-	Si	No	Si	No	No	0	-	-
BNT	Matlab/C	Si	W,U,M	G	No	Si	Si	Si	0	D,U	Muchos
BNJ	Java	-	-	D	Si	No	Si	No	0	D	JTree, IS
BNL	Matlab	-	-	D	No	No	No	No	0	D	JTree
BUGS	No	No	W,U	Cs	Si	Si	No	No	0	D	G
Causal Discoverer	No	No	W	-	-	No	Si	No	0	D	-
Clspace	Java	No	W,U	D	Si	No	No	No	0	D	Varelim
CRFtoolbox	Matlab/C	Si	-	No	No	Si	No	No	0	U	-
DBNbox	Matlab	-	-	Si	No	Si	No	No	$	D	Varios
Derivelt	No	-	-	-	-	Si	Si	-	$	D	JTree, Gibbs
Elvira	Java	Si	W,U,M	Cd,Cx	Si	Si	Si	Si	0	D	JTree, varelim, Is
GDAGsim	C	Si	W,U,M	G	No	No	No	No	0	D	-
GeNle and SMILE	SMILE	Si	W,U,M	Cs	Si	Si	Si	Si	0	D	JTree, IS
GMRFSim	C	Si	W,U,M	G	No	No	No	No	0	U	-
GMTk	No	Si	U	D	No	Si	Si	No	0	D	JTree
gR	-	-	-	-	-	-	-	-	0	-	-
Grappa	-	-	-	D	No	No	No	No	0	D	JTree
HdBCS	C++	-	-	G	No	Si	Si	No	0	U	SL
Hugin Expert	N	Si	W	G	Si	Si	CI	Si	$	CG	JTree
Hydra	Java	-	-	Cs	Si	Si	No	No	0	U,D	-
Infer.NET	C#	Si	Si	Si	No	Si	No	No	0	Si	VMP, EP, G
JAGS	Java	Si	-	Si	No	Si	No	No	0	Si	G

[12] Se han eliminado de la tabla los programas que no se han podido verificar sus propiedades.

Nombre	Src	API	Exec	Cts	GUI	Params	Struct	Utility	Free	Undir	Inference
Java Bayes	Java	Si	W,U,M	D	Si	No	No	Si	0	D	Varelim, JTree
LibB	No	Si	W	D	No	Si	Si	No	0	D	SL
MIM	No	No	W	G	Si	Si	Si	No	$	CG	JTree
Mocapy++	C++	Si	W,U,M	G	No	Si	No	No	0	D	G
MSBNx	No	Si	W	D	Si	No	No	Si	0	D	JTree
Netica	No	Si	W,U,M	G	Si	Si	No	Si	$	D	JTree
Bayes net Learner	No	No	W,U	D	No	Si	Si	No	0	D	SL
PMT	Matlab/C	-	-	D	No	Si	No	No	0	D	-
PNL	C++	-	-	D	No	Si	Si	No	0	U,D	JTree
Pulcinella	Lisp	Si	W,U,M	D	Si	No	No	No	0	D	-
RISO	Java	Si	W,U,M	G	Si	No	No	No	0	D	Polytree
Sam lam	No	No	W,U	G	Si	Si	No	Si	0	D	-
LADR	No	Si	-	Cd	No	No	Si	No	$	D	None
Tetrad	No	No	W,U	G	No	Si	CI	No	0	U,D	SL
UC Irvine	Si	No	W,U	D	No	No	No	No	0	U,D	-
UnBBayes	Java	-	C	D	Si	No	Si	No	0	D	JTree
Vides	Java	Si	W,U	Cx	Si	Si	No	No	0	D	VMP
WinMine	No	No	W	Cx	Si	Si	Si	No	0	U,D,	SL
XBAIES 2.0	No	No	W	G	Si	Si	No	Si	0	CG	JTree

Anexo B: Manual de instalación

A continuación, se mostrarán los pasos a seguir para la instalación de Netica para su utilización en el proyecto. Se señalarán las configuraciones necesarias para el entorno de desarrollo y todos aquellos aspectos más relevantes que influyen en el correcto funcionamiento del sistema.

El presente manual distinguirá por un lado, la instalación de la aplicación Netica como interfaz de usuario y por otro lado, la incorporación de la librería necesaria para el desarrollo del proyecto. Y como último apartado, se expondrá el proceso de instalación de la interfaz de usuario.

Aplicación Netica

Netica es un entorno gráfico que contiene un conjunto de funciones para la descripción y la inferencia de redes bayesianas que se pueden utilizar para desarrollar aplicaciones compatibles con entornos de Microsoft Windows.

Su interfaz es bastante intuitiva y amigable, además de que con ella es posible realizar una abducción total y/o parcial, mediante forma gráfica o textual. Se pueden realizar análisis de sensibilidades y permite en uso de variables continuas. Otra característica es que se permiten definir submodelos además de poder visualizar los resultados de los estados de cada nodo sobre el propio grafo de la red mediante diagramas de barras.

El conjunto completo de características de la aplicación Netica se pueden consultar en la página de Norsys Software Corp.[13], que es la que distribuye en programa.

En nuestro caso particular se trabajará con la última versión disponible “Netica v4.09”[14] cuya versión será la que se empleé.

Para la instalación del software será necesario realizar las siguientes instrucciones:

1. Descargar[15] Netica de la página de Norsys, en nuestro caso se empleará la versión para Microsoft Windows (95/ 98/ NT4/ 2000/ XP/ Vista), aunque se encuentra también una versión disponible para Macintosh OSX.

2. Tras la descarga, simplemente se hará doble click en el icono del archivo y se realizará automáticamente la auto-extracción. Tras esto, se le preguntará que directorio se utilizará para la instalación de los archivos. La imagen B.1. muestra dicho paso a realizar.

[13] La descripción general está disponible en http://www.norsys.com/netica.html.
[14] La descripción de las características específicas de cada versión se pueden obtener de http://www.norsys.com/new_features_app.htm.
[15] La versión actual de Netica se puede obtener en la dirección http://www.norsys.com/downloads/Netica_Win.exe

Imagen B.1. Auto-extracción

3. Tras la auto-extracción se hará doble click en el icono "Netica.exe", localizado en el directorio de la extracción, para la ejecución de la aplicación. Al trabajar sin derechos de administrador y no poseer licencia del programa, solamente se podrán guardar redes con 15 nodos como máximo y no se podrá realizar un aprendizaje paramétrico de la red con más de 1000 casos posibles. La imagen B.2 muestra la interfaz de la aplicación Netica tras su ejecución.

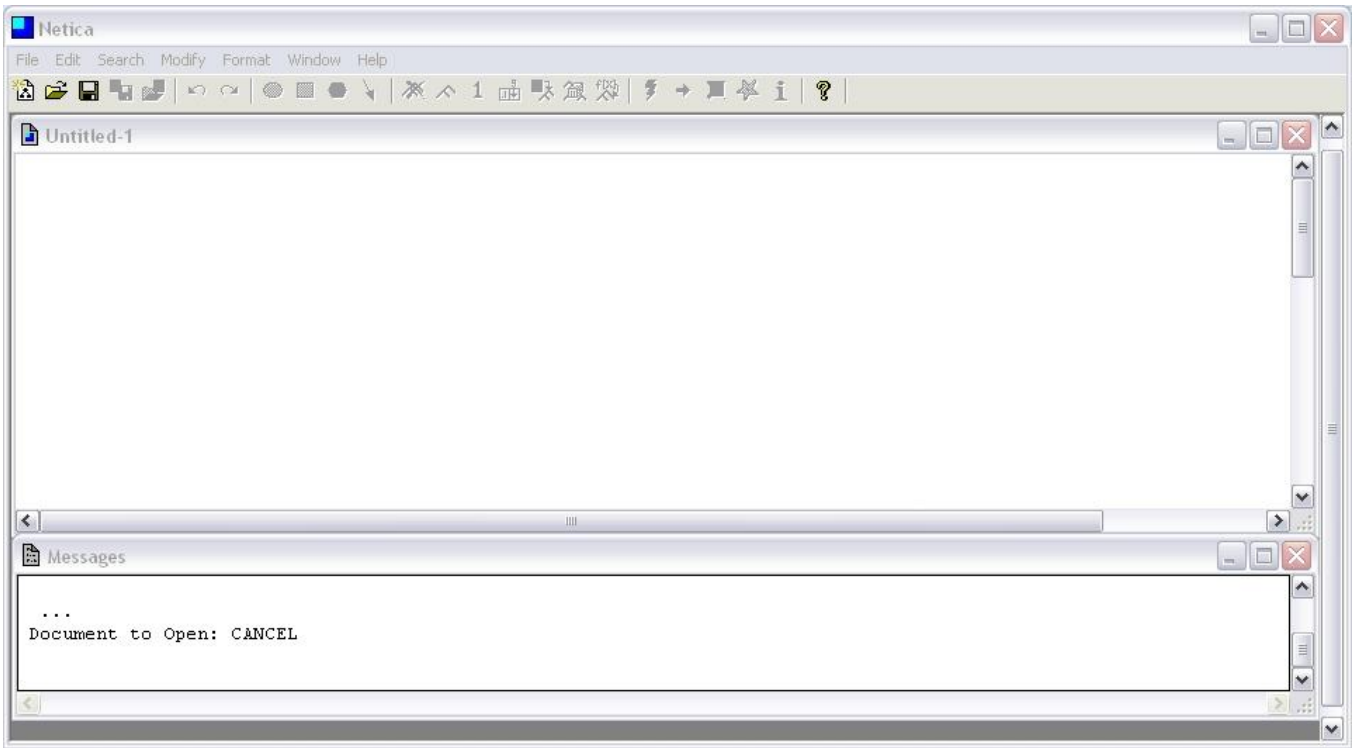

Imagen B.2. Interfaz de usuario de Netica

Netica API

Para la instalación del conjunto de librerías necesarias en el desarrollo de la aplicación dentro del entorno de programación de Microsoft Visual Studio, será necesario haber realizado previamente el proceso de instalación de la aplicación Netica. Tras este paso únicamente será necesario agregar la referencia a la aplicación Netica.exe.

El primer paso, consiste en acceder a la página de propiedades del proyecto y en la opción de propiedades comunes seleccionar agregar nueva referencia. Tras la realización de las fase anterior, se nos mostrará una venta como la mostrada por la imagen B.3., en la que habrá que seleccionar la pestaña COM y buscar el nombre del componente "*Netica 4.09 Object Library*"; tras lo cual se agregará dicho componente a la solución del proyecto y a partir de entones se podrán utilizar las clases que en él se implementan.

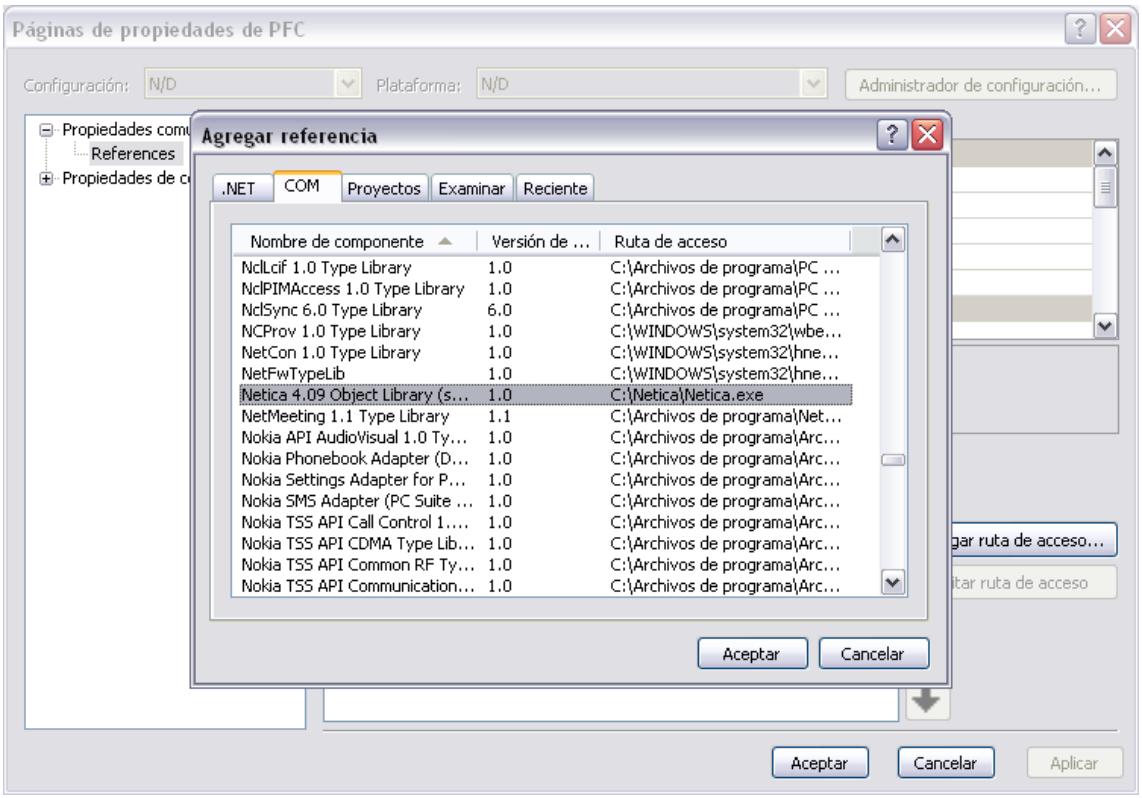

Imagen B.3. Agregación de la referencia a Netica API

Interfaz desarrollada

En este punto se describirá el proceso de instalación de la interfaz de usuario para el uso de redes bayesianas dinámicas. Es necesario haber realizado previamente el proceso de instalación de la aplicación Netica recogido al principio de éste anexo. Esto es debido a que dicho proceso nos establecerá en el sistema operativo la relación entre las extensiones de los ficheros de las redes bayesiana y el programa correspondiente para su ejecución además de incorporar la API al sistema y permitirnos incorporarla en distintos proyectos de Visual Studio.
En el proceso de instalación de la interfaz de usuario bastará con ejecutar el archivo llamado “Setup.exe” o “Setup.msi” para posteriormente seguir el asistente.

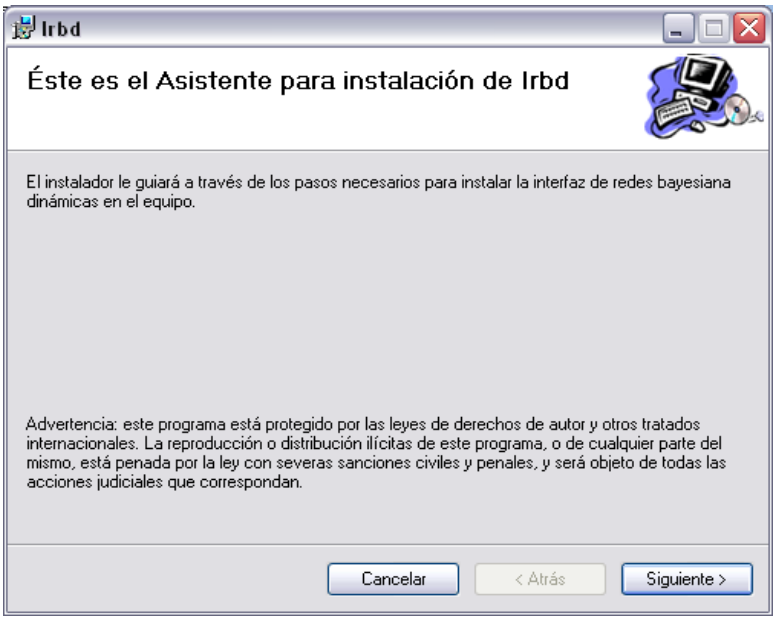

Imagen B.4. Primer paso de instalación de la interfaz desarrollada

Una vez iniciado el proceso de instalación, se pulsará siguiente para proceder a la siguiente ventana donde se nos pedirá el directorio de instalación además de solicitar que usuarios podrán utilizar el programa.

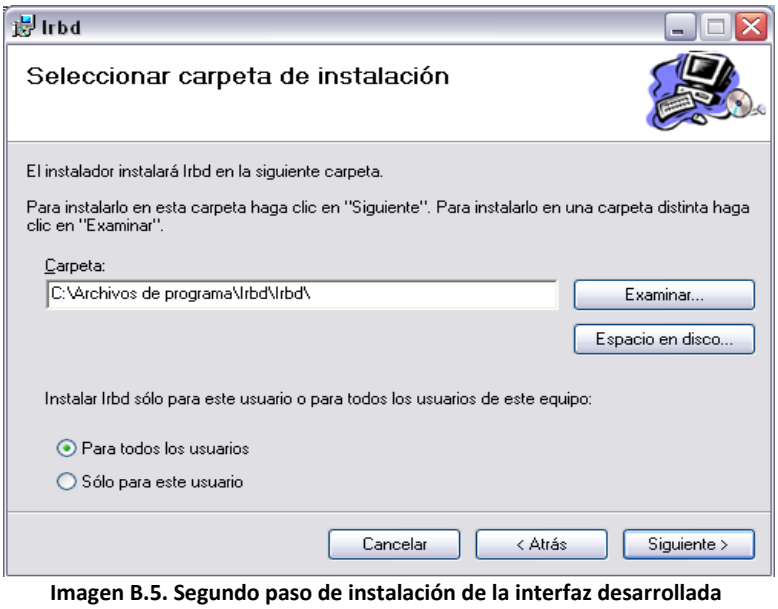

Imagen B.5. Segundo paso de instalación de la interfaz desarrollada

Una vez completado el proceso anterior pulsaremos el botón siguiente ocasionando que aparezca una nueva ventana que nos pedirá confirmar la instalación tras cuál pulsaremos sobre el botón siguiente para confirmar.

Por último, una vez finalizada la instalación, se pulsará sobre el botón cerrar; si el proceso ha ido correctamente se tendrá que haber creado en la ruta de instalación los ficheros necesarios para la ejecución de la interfaz así como un acceso directo llamado “Irbd” en el escritorio de Windows. Haciendo doble click sobre el acceso directo, se ejecutará el programa y se podrá empezar a trabajar con él.

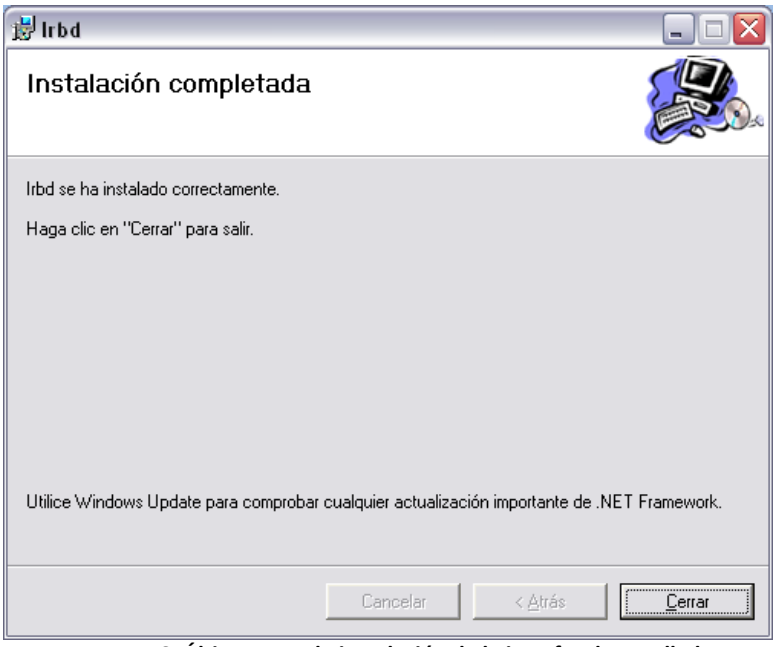

Imagen B.6. Último paso de instalación de la interfaz desarrollada

Anexo C: Manual de usuario

Este anexo recoge el manual de usuario de la aplicación interfaz desarrollada así como de la aplicación Netica. El objetivo no es otro que el de dar al usuario una pequeña visión del conjunto de funciones disponibles, de las opciones habilitadas y de la utilización correcta de estas.

Interfaz desarrollada

La imagen C.1. Diseño de la interfaz (ver página siguiente), presenta la visión común de la interfaz en la que se han enumerado las distintas áreas que se pueden observar en el formulario de la aplicación. El objetivo es el de poder dar una definición a los distintos controles/botones que posteriormente se mencionen en el presente manual.

A continuación se presentan las distintas áreas y los controles asociados de izquierda a derecha de la imagen C.1. Diseño de la interfaz gráfica (ver página siguiente).

1. Barra de acciones, sus controles son:
   - Iniciar sistema, permite mostrar la red bayesiana a través de la aplicación Netica para su posterior utilización.
   - Detener sistema: detiene la aplicación volviendo al inicio del sistema.
   - Número de cortes de tiempo: permite introducir un valor numérico para la realización de varios cortes de tiempo. Para establecer el valor será necesario confirmarlo pulsando el botón “enter”.
   - Siguiente corte de tiempo: permite realizar la simulación de un corte de tiempo.
   - Guardar gráfico: permite salvaguardar un gráfico con el estado de las variables de observación.
   - Frecuencia de lectura del fichero de entrada (segundos): es un nuevo control que aparecerá en el caso de que exista un fichero de entrada (presente en la imagen C.1. con el valor 5) que permitirá modificar la frecuencia de lectura del fichero de datos. Para establecer el valor será necesario confirmarlo pulsando el botón “enter”.
   - Tiempo total: representado por la última casilla de esta área, tiene como objetivo, informar sobre el paso del tiempo a lo largo del proceso de simulación desde que se inicio el sistema.
2. Datos del fichero de entrada, compuesto por:
   - Ruta del fichero de datos: muestra la localización física del fichero de datos.
   - Botón examinar: permite buscar el fichero de datos a través de una ventana de exploración.
3. Datos de la red bayesiana, compuesto por:
   - Ruta del la red bayesiana: muestra la localización física de la red bayesiana.
   - Botón examinar: permite buscar la red bayesiana de datos a través de una ventana de exploración.
4. Selección de relaciones temporales, área que notifica las relaciones existentes en la red bayesiana y permitiendo seleccionar aquellas que tengan una relación temporal.

5. Selección de variables de observación, área que notifica las variables presente en la red bayesiana permitiendo ser seleccionadas para su posterior estudio gráfico.
6. Control de progreso, compuesto por:
   - Botón de detener/iniciar proceso: este control tiene una función triple. Si el sistema está en modo manual, permite detener la simulación de inferencia si el sistema se demora. En cambio si es sistema está en modo automático, el botón puede detener o iniciar este modo.
   - Barra de progreso: representación visual del tiempo que resta para completar la simulación de varios cortes de tiempo.

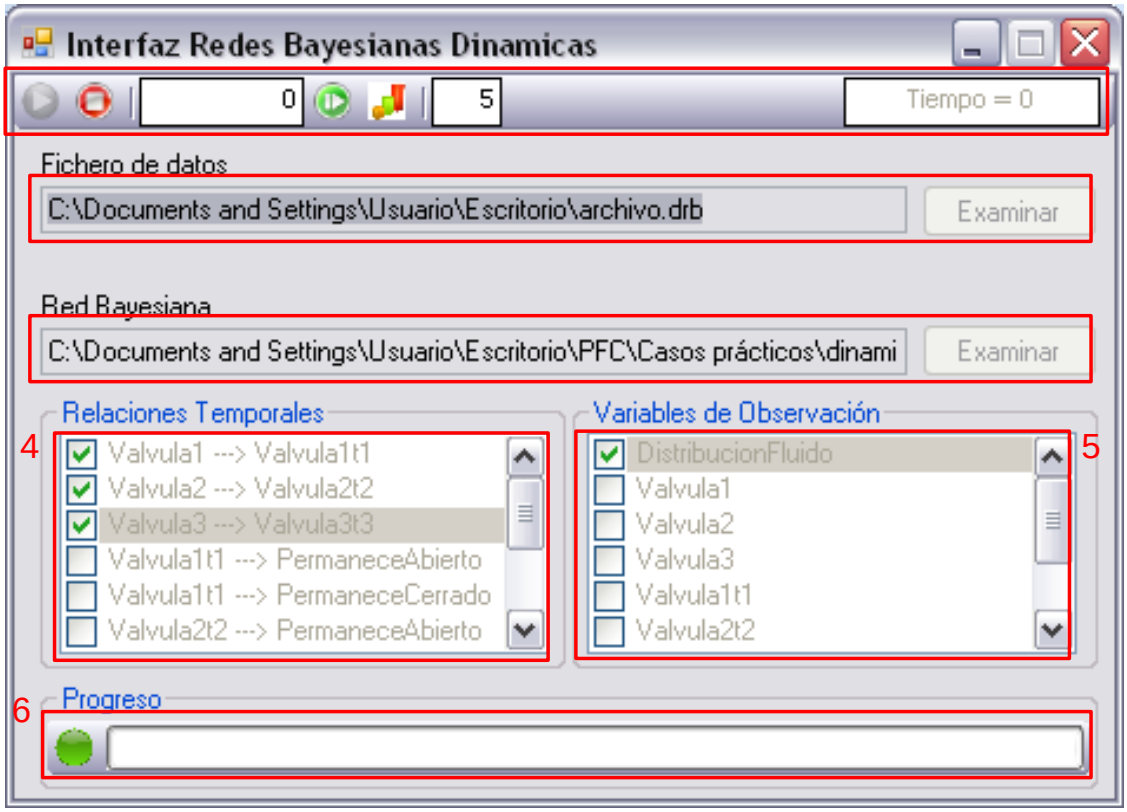

Imagen C.1. Diseño de la interfaz gráfica

En los dos próximos apartados se mostrará el manejo del sistema para los dos tipos de simulación existentes, la manual y la automática. Para dichos procesos se ha supuesto que la red bayesiana dinámica ya ha sido construida a través de la aplicación Netica y solamente queda realizar el estudio sobre la evolución de la misma a través de la interfaz desarrollada.

Simulación manual

La simulación manual nos permitirá ir introduciendo los hallazgos según éstos se presenten en la red bayesiana a la largo del tiempo. Para visualizar más fácilmente los posibles estados en los que nos podemos encontrar tras la realización de las posibles secuencias de acciones del usuario, se muestra en la figura C.1. el diagrama de estados del modo manual.

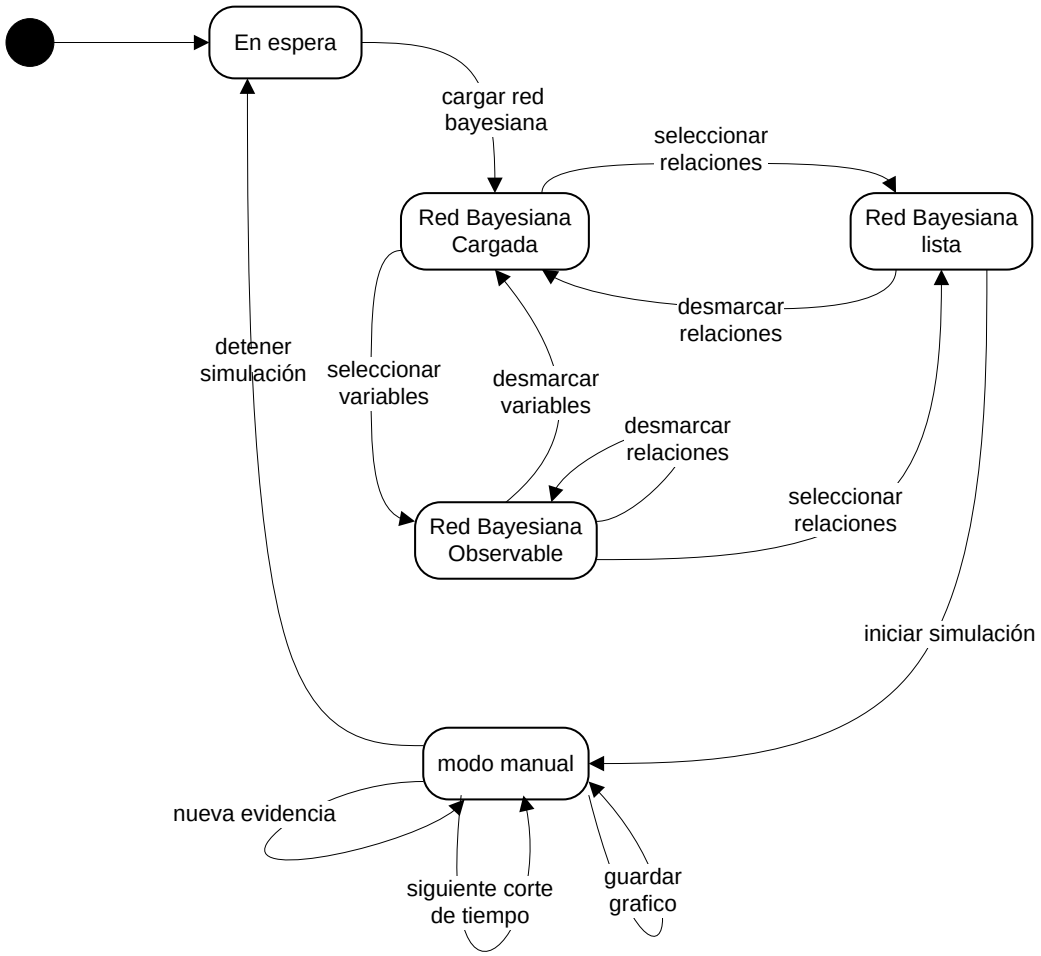

Figura C.1.Diagrama de estados del modo manual

Los pasos comunes a seguir por el usuario para la simulación manual son:

1. Ejecutar la interfaz haciendo doble click sobre el programa “Irbd.exe”.
2. Realizar la carga de la red bayesiana presionando sobre el botón examinar del área de datos de la red bayesiana y posteriormente seleccionando la ubicación física de la red.
3. Seleccionar las relaciones temporales que existan en el modelo marcando o desmarcando las relaciones del área número 3.
4. En caso de querer observar alguna variable para su posterior representación gráfica se marcarán aquellas de interés que se quieras visualizar. Este paso es opcional y no es imprescindible para la simulación manual. En el caso de realizar

este paso, la precedencia respecto al paso 3 es meramente casuístico, ya que no existe un orden preestablecido.

5. Se inicia el proceso de simulación apretando sobre el botón iniciar.

La imagen C.2. muestra el estado de la aplicación tras realizar el paso número 5. Como se puede apreciar se inhabilitarán aquellas acciones que no tienen ningún sentido realizar y por el contrario se activan aquellas que se puedan realizar.

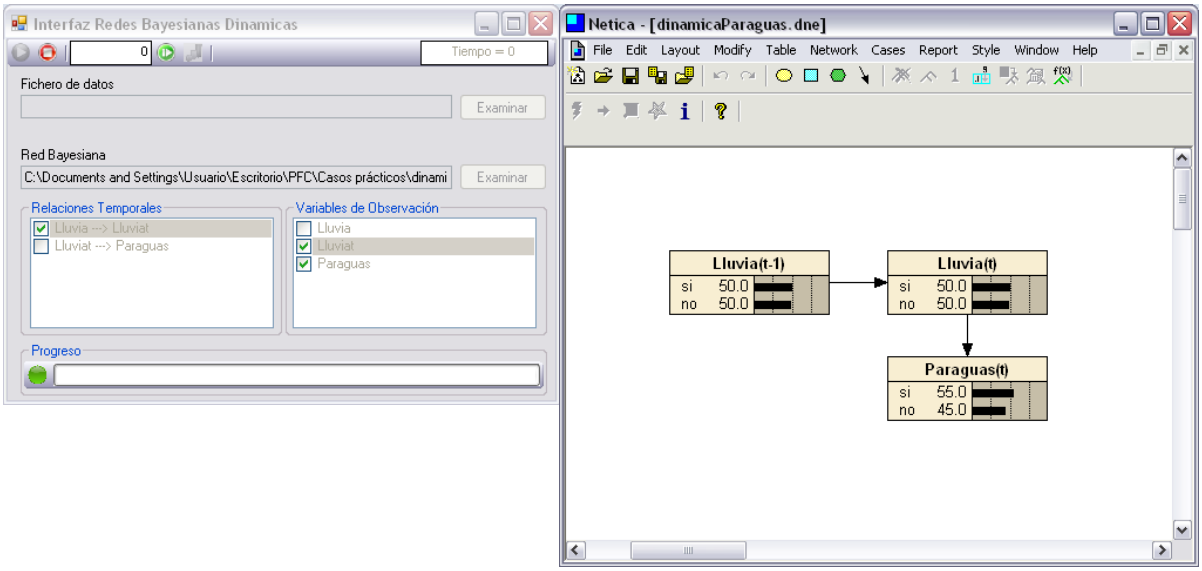

Imagen C.2. Visión del modo manual tras su iniciación

Los siguientes pasos pueden realizar en cualquier orden ya que no existe ninguna precedencia. La única excepción es que para poder salvaguardar un gráfico, además de haber seleccionado previamente alguna variable de observación, se tiene que haber realizado al menos, la ejecución de un corte de tiempo. La secuencia es:

6. Incorporar la o las evidencias en la red bayesiana de la aplicación Netica de la variables de observación haciendo click encima del estado que se quiere evidenciar.
7. Pulsar sobre el botón siguiente corte de tiempo para realizar el proceso de inferencia saltando una unidad en el tiempo.
8. Si se quiere realizar la simulación de varios cortes de tiempos se introducirá el valor numérico en la casilla de número de cortes de tiempo y se pulsará “enter”, haciendo que automáticamente se realice dicho número de cortes.
   8.1. Si se pulsa sobre el botón detener/iniciar proceso del área número seis, la ejecución de los cortes de tiempo se detiene por el valor en el que fuese.
9. Si se quiere salvaguardar el gráfico generado de pulsará sobre el botón de guardar gráfico procediendo a elegir la ruta de destino y el nombre del fichero.
10. Por último, una vez que se ha terminado el proceso de simulación manual se puede detener el sistema presionando sobre el botón detener y volviendo al estado inicial tras la ejecución de la aplicación.

La imagen C.3. presenta el instante del estado de ejecución de la aplicación en el momento t = 94 en el que se ha introducido un valor de mil cortes de tiempo, antes de ser detenido dicho proceso.

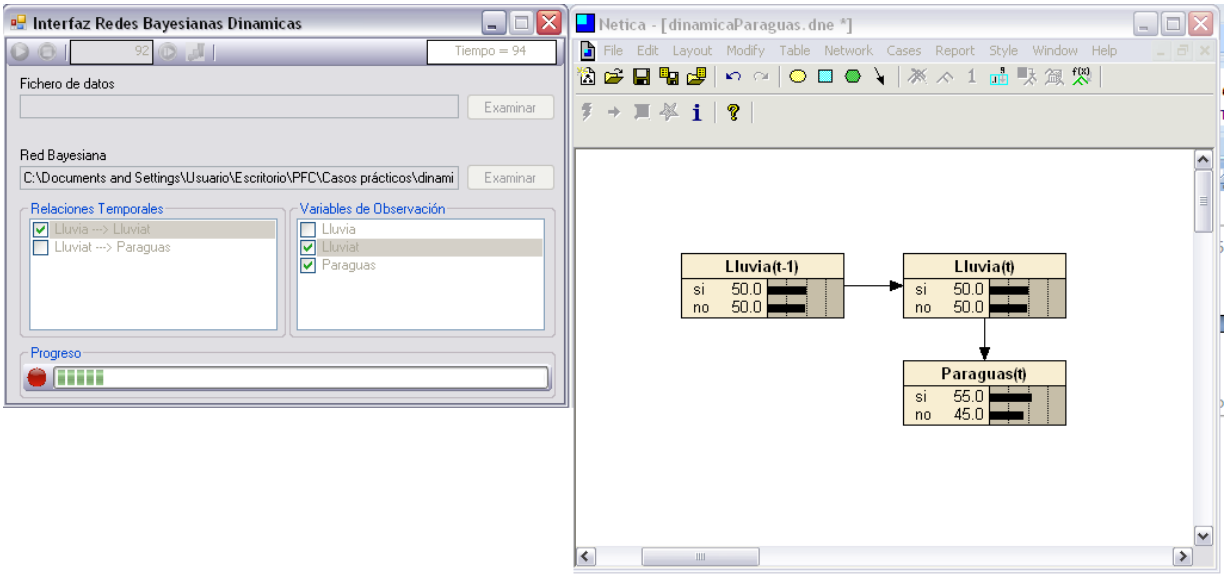

Imagen C.3. Visión del estado manual tras varios cortes de tiempo.

La imagen C.4. vemos la salida del grafo correspondiente al caso de simulación anterior donde observamos que para el instante de tiempo t = 1, la presencia del paraguas se hizo evidente.

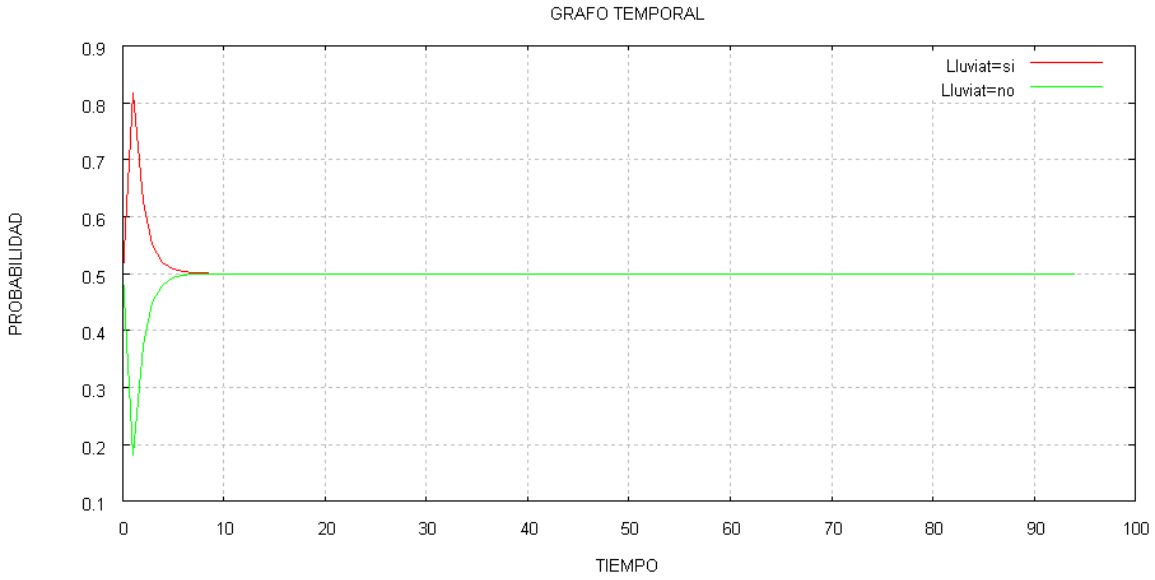

Imagen C.4. Grafo temporal del modo manual tras la ejecución.

Simulación Automática

La simulación automática está pensada para ser utilizada de modo simultáneo con el uso de sistemas expertos con incertidumbre que realicen las fases de extracción de características obteniendo las evidencias en tiempo real de las variables del modelo. Esto nos permite ver en tiempo real la evolución de las probabilidades de las variables en la red bayesiana dinámica observando los distintos hallazgos que se producen en el sistema experto. Para realizar esta simulación, el sistema experto externo deberá de presentar los hallazgos en un fichero externo siendo éste el medio de comunicación entre la interfaz de redes bayesiana dinámicas y el sistema experto.

Para visualizar más fácilmente los posibles pasos en los que nos podemos encontrar tras la realización de la secuencia de uso del modo automático por parte de las acciones del usuario, se muestra en la figura C.2. el diagrama de estados correspondiente. Se han eliminado ciertos estados para no incrementar la complejidad debido a que ciertas acciones del usuario se pueden realizar sin precedencias como es el caso de cargar el fichero de datos antes o después de la red bayesiana. El estado "modo manual" se corresponde con el del proceso

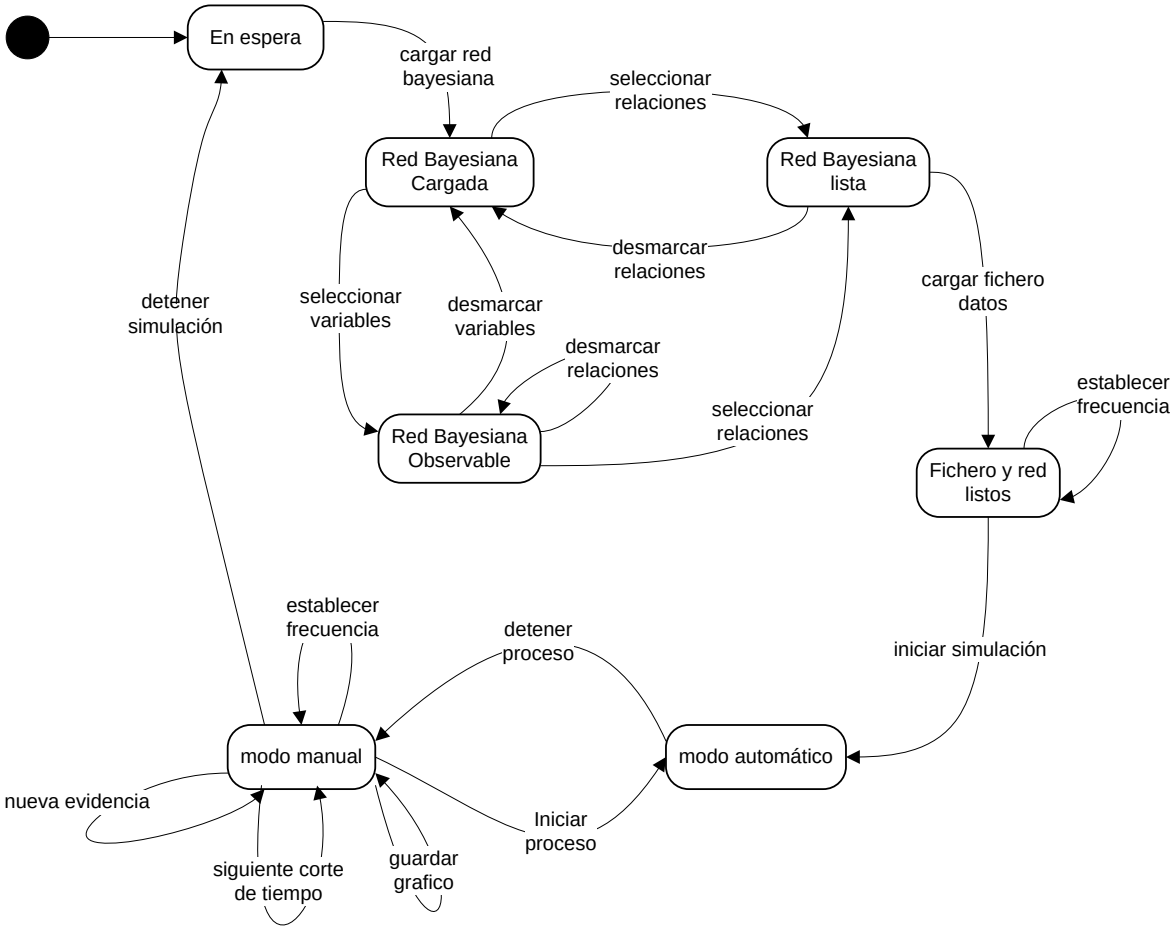

Figura C.2.Diagrama de estados del modo automático

Los pasos comunes a seguir por el usuario para la simulación manual son:

1. Ejecutar la interfaz haciendo doble click sobre el programa “Irbd.exe”.
2. Realizar la carga de la red bayesiana presionando sobre el botón examinar del área de datos de la red bayesiana y posteriormente seleccionando la ubicación física de la red.
3. Seleccionar las relaciones temporales que existan en el modelo marcando o desmarcando las relaciones del área número 3.
4. En caso de querer observar alguna variable para su posterior representación gráfica se marcarán aquellas de interés que se quieras visualizar. Este paso es opcional y no es imprescindible para la simulación manual. En el caso de realizar este paso, la precedencia respecto al paso 3 es meramente casuístico, ya que no existe un orden preestablecido.
5. Se cargará el fichero de datos presionando sobre el botón examinar del área de datos del fichero de entrada (área 2) y posteriormente se seleccionará la ubicación del fichero.
6. En caso de querer modificarse la frecuencia con la que se va a proceder a la lectura del fichero de datos, se introducirá un valor numérico en la casilla frecuencia de lectura del fichero de entradas (segundos) y se presionará el botón “enter” para establecerse el nuevo valor.
7. Se inicia el proceso de simulación automático apretando sobre el botón iniciar.

La imagen C.5. muestra el estado de la aplicación tras realizar el paso número 7. Como se puede apreciar se inhabilitarán todas las acciones ya que el usuario no puede realizar un uso manual de la aplicación hasta que no detenga el proceso de simulación automático pulsando sobre el botón de detener/iniciar proceso.

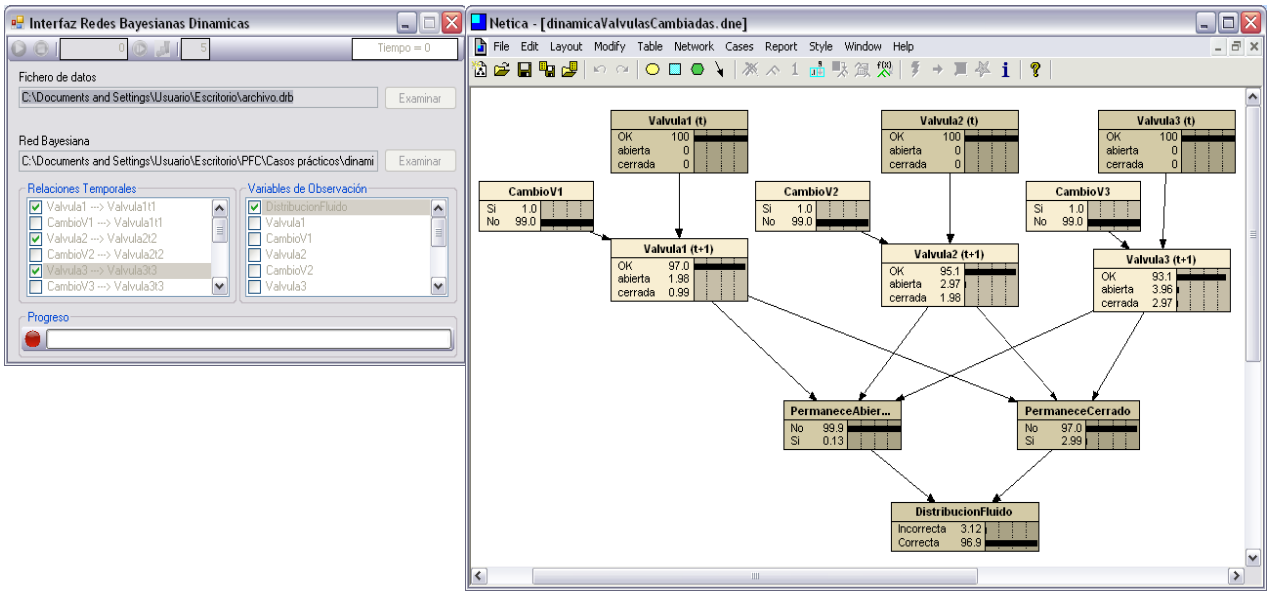

Imagen C.5. Visión del modo automático tras su iniciación

La ejecución automática realizará la simulación de cortes de tiempo en la red bayesiana con una frecuencia igual al valor preestablecido antes del inicio del modo, que como se ha podido apreciar de la imagen anterior tiene un valor igual a 5 segundos. Por consiguiente antes de realizar el corte de tiempo del proceso automático, la interfaz de redes bayesianas dinámicas se encargará de leer el fichero de entrada de datos actualizando los hallazgos que se encuentre en las variables de la red.

El botón de detener/iniciar proceso nos permite cambiar rápidamente entre el modo automático y el modo manual permitiendo en éste último caso realizar las acciones de usuario anteriormente descritas en el proceso de simulación manual.

La imagen C.6. presenta el instante del estado de ejecución de la aplicación en t = 18 tras la cual se detuvo el proceso automático.

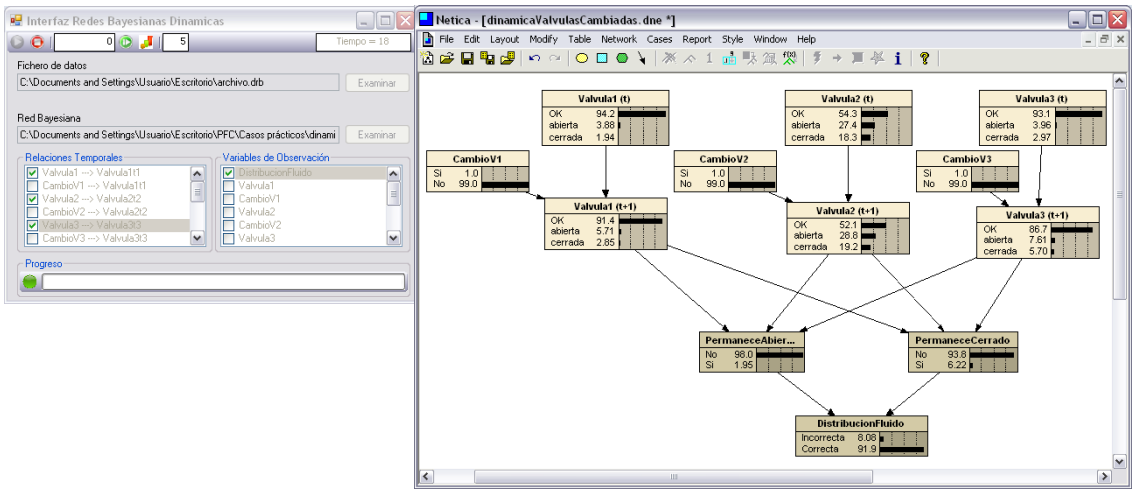

Imagen C.6. Visión del estado automático tras varios cortes de tiempo.

Por su parte la imagen C.7. presenta la salida del proceso automático correspondiente al caso de simulación anterior donde las evidencias han venido dadas por el fichero de datos de entrada.

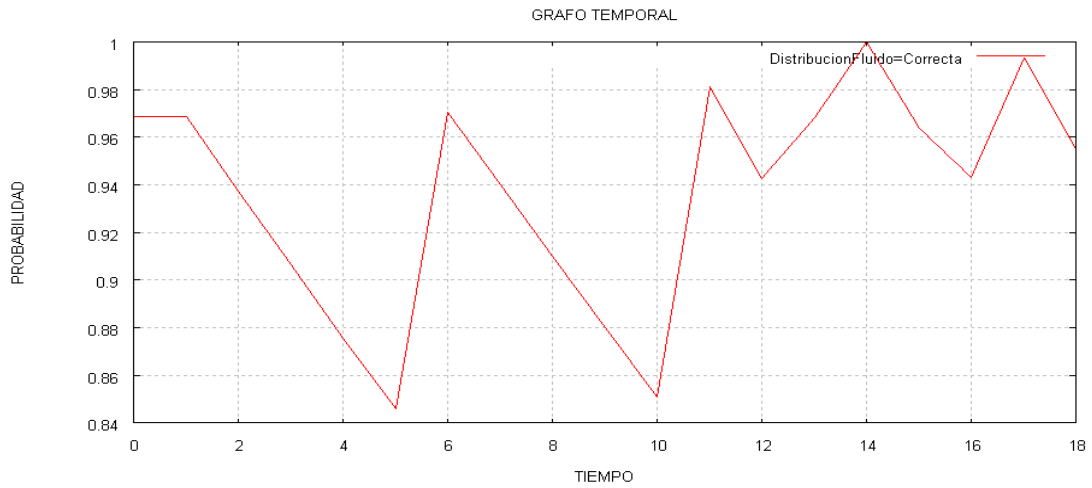

Imagen C.7. Grafo temporal del modo automático tras la ejecución.

Como ya se ha mencionado anteriormente, la simulación automática nos permite conectar la red bayesiana, a través del fichero de entrada de datos con extensión ".drb" (ver apartado 5.4.1. Fichero de datos), con otro sistema paralelo de tal manera que nos permita crear un sistema experto capaz de realizar el proceso de fusión 2, de tal manera que se produzca una evaluación de la situación a través de las evidencias presentadas.

El fichero de datos, que es la interfaz de entrada de evidencias de la red bayesiana dinámica, es a su vez el fichero de salida del sistema paralelo encargado del nivel 1 de fusión de tal manera que conecte la fuente de datos con el sistema desarrollado. Estas fuentes de datos pueden ser variadas, como datos de sensores o procesados de videocámaras o similares, donde solamente se necesita que el formato del fichero datos sea correspondiente a la entrada del entorno de usuario. De esta forma se consigue desvincular el nivel 1 y el nivel 2 de fusión dentro de un único sistema.
La figura C.3. representa el esquema de cómo podrían conectarse la fuentes de datos a través de los dos niveles de fusión. La interfaz intermedia entre los dos niveles, vendrá representada por el fichero de datos ".drb" que contendrá las evidencias producidas de las variables de la red bayesiana dinámicas en el formato especificado en el apartado 5.4.1. Es por tanto, función del sistema de procesamiento realizar las acciones necesarias desde las fuentes de datos para trasmitir las evidencias al entorno de usuario que aplica el uso de las redes bayesianas dinámicas al análisis de situaciones, a través de la escritura del fichero de datos.
El entorno de usuario, encargado del nivel 2, realizará la lectura temporal del fichero de datos para así poder transmitir los valores evidenciados en las variables de la red y realizar el proceso de inferencia bayesiana.

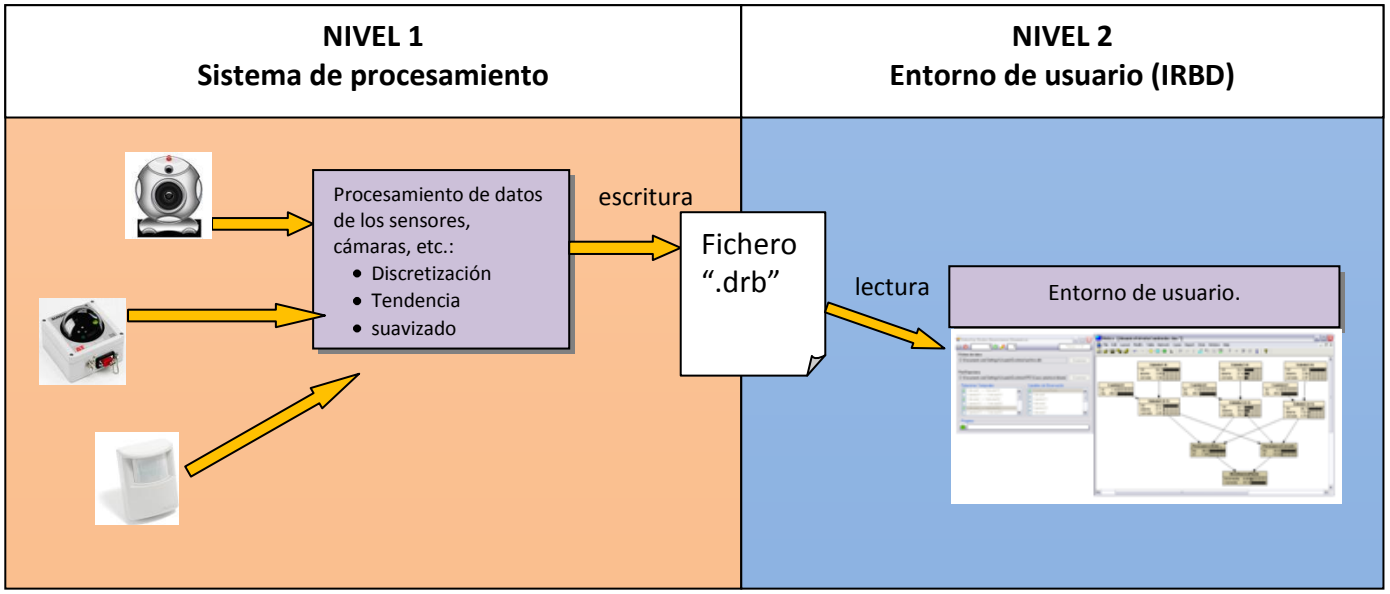

Figura C.3. Representación de la conexión entre el nivel 1 y el nivel 2 de fusión.

Aplicación Netica

A continuación se presenta un breve manual de uso de la aplicación Netica. La intención es ofrecer al usuario el conjunto de opciones más habituales a la hora de realizar la construcción de la red bayesiana, ya que una explicación más detallada de todas las opciones que el usuario puede realizar en la interfaz se pueden consultar en el manual online (57) y en la guía de usuario (58) ofrecidos por Norsys Software Corporation.

Netica es una herramienta creada para trabajar con redes de creencia y con diagramas de influencia. Proporciona una interfaz intuitiva que permite agilizar el proceso de construcción de las redes y las relaciones entre las variables, permitiendo incorporar las probabilidades a priori o condicionales, en forma de ecuaciones, o extraídas de archivos de datos. Además una vez creada la red se puede llevar a cabo el proceso de inferencia a través del conocimiento introducido en forma de hallazgos haciendo que Netica encuentre las probabilidades para todas las variables desconocidas.

De entre sus características más destacadas se pueden señalar:

- Representación visual de la red y de sus enlaces.
- Permite realizar la propagación del razonamiento probabilístico de manera rápida y eficaz en redes cíclicas o multiconectadas.
- Permite almacenar y cargar las redes creadas por el usuario.
- Permite la utilización de archivos de casos para probar el funcionamiento de una red, permitiendo obtener información sobre las matrices de confusión, las tasas de error logarítmicas y cuadráticas, etc. para cada nodo deseado.
- Puede aprender las relaciones probabilísticas entre los datos.
- Permite una edición fácil, con multitud de opciones de usuario para copiar, cortar, duplicar, pegar, etc. los nodos y enlaces de las redes.
- Contiene una amplia ayuda para prevención de errores notificando por pantalla las posibles incoherencias que se puedan introducir, como puede ser una entrada inválida en de cantidades incorrectas en las tablas de probabilidad condicionada. Además de aporta niveles ilimitados de deshacer y/o rehacer acciones.
- Incorpora herramientas para la discretización fácil de las variables continuas que se presenten en el modelo.
- No existe un límite en cuanto al tamaño o la complejidad de la red, la construcción solamente vendrá limitada por la cantidad de memoria disponible en el sistema.
- Empleo de distintos algoritmos de inferencia que permitan mejorar el proceso de inferencia.
- API desarrollada para un amplio conjunto de lenguajes de programación que permiten el empleo de las funciones de Netica desde programa externos.

Como se puede ver la aplicación Netica contempla un gran conjunto de funcionalidades que nos permiten seleccionarla como la herramienta a emplear.

Recordemos que Netica es una software corporativo por lo que se trabajará con la versión de prueba que nos permite un número limitado de variables en la construcción de la red.
A continuación presentamos los pasos para la construcción de una red bayesiana a través de la aplicación Netica, eligiendo como caso de ejemplo el primer caso práctico presentado en el apartado 6.1. El primer paso es crear una red en blanco a través de la secuencia File→Network. Tras este paso la incorporación de nodos a la red, que contendrán las variables del modelo, se realiza apretando sobre el botón y posteriormente haciendo click sobre la red en blanco. Para incorporar las relaciones entre las variable habrá que seleccionar el icono y presionar posteriormente sobre los dos nodos que se quieran relacionar.
La imagen C.8. presenta esta serie de pasos realizados. Para guardar la red bayesiana bastará con pulsar sobre el icono o realizar la secuencia File→Save.

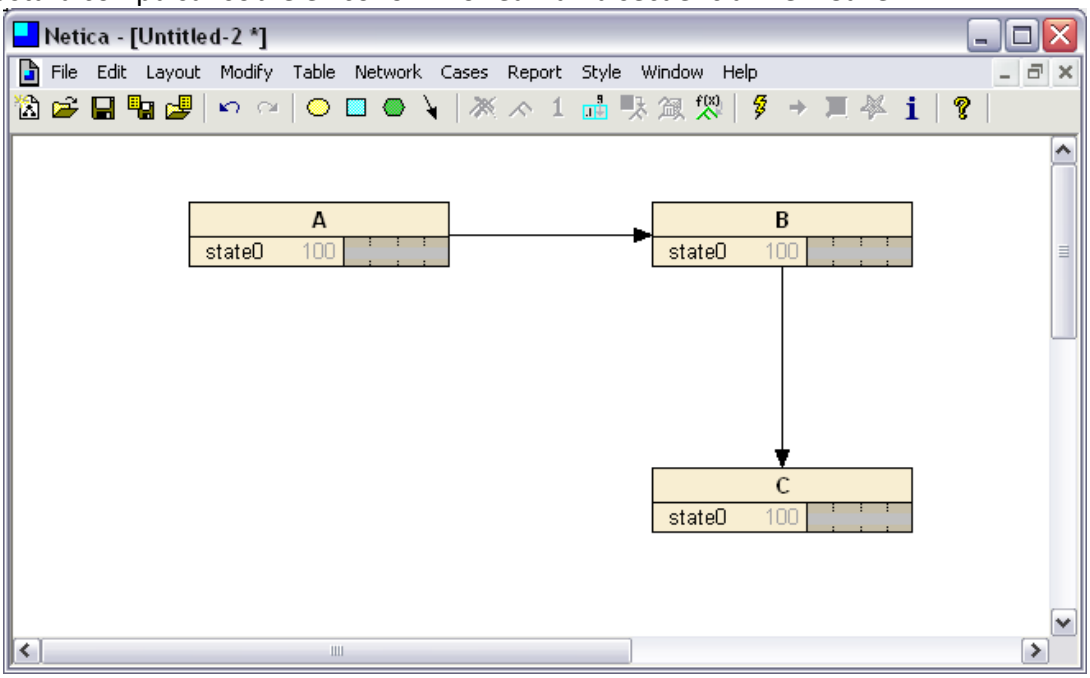

Imagen C.8. Incorporación de nodos y enlaces.

Para poder acceder a las opciones del nodo y poder introducir información sobre las características del mismo, primero se seleccionará el nodo y posteriormente con el botón derecho del ratón se desplegará el menú de opciones del nodo. La imagen C.9. presenta este proceso. La opción "Properties" permite definir características del nodo como son el tipo de dato que va a contener, el nombre, la descripción, el conjunto de estados, etc.

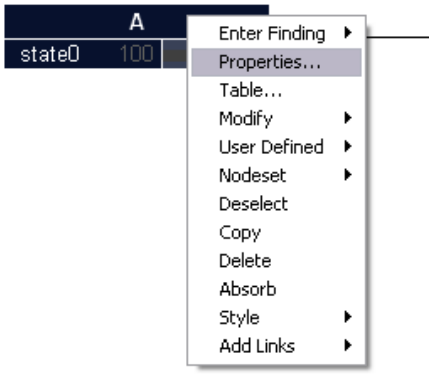

Imagen C.9. Visión de las opciones de un nodo.

La opción "Table…" permite acceder a las tablas de probabilidad condicionadas del nodo y completarlas insertando su valor numérico.

La imagen C.10. presenta la tabla de propiedades y la tabla de probabilidades de un nodo relativas a las acciones de “Properties” y “Table..”.

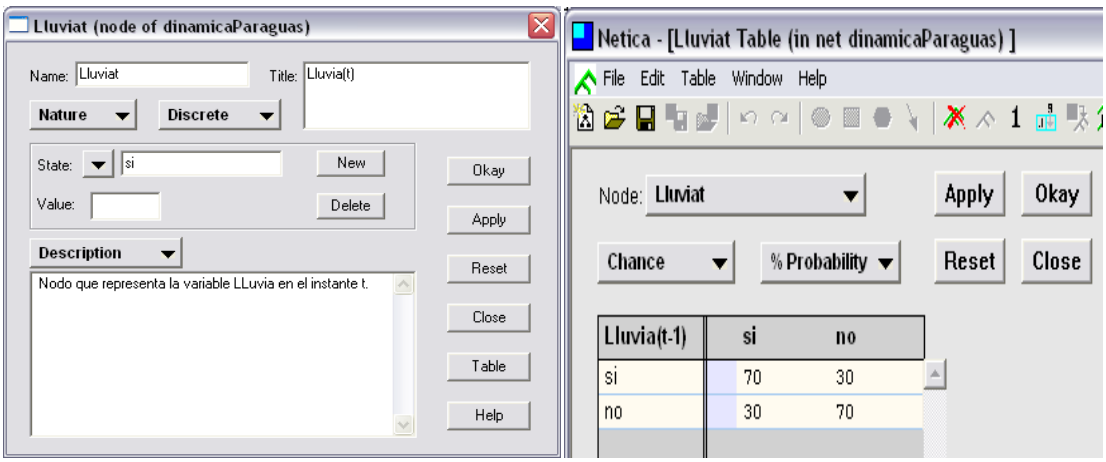

Imagen C.10. Tabla de propiedades y tabla de probabilidades de un nodo.

Tras la incorporación de la información de las variables y la asignación de los valores numéricos de las tablas de probabilidad, la red estaría disponible para realizar el proceso de inferencia. Para entrar en este modo de ejecución, previamente habrá que compilar la red bayesiana, que consiste en realizar la propagación inicial de las probabilidades de las variables por la red antes de la incorporación de un hallazgo. Para ello se pulsa sobre el botón  o se realiza la secuencia Network→Compile.

En el proceso de inferencia cualquier nuevo hallazgo será introducido en la red pinchando sobre el estado de la variable que se ha evidenciado y automáticamente se realizará la actualización de las probabilidades del resto de variable (siempre que esté activa la opción Network→Automatic Updating). Por último la eliminación de los casos introducidos puede realizarse pinchando sobre el botón  o realizando la secuencia Network→Remove Findings.

La imagen C.11. muestra la presentación de la aplicación Netica tras la incorporación de la evidencia “Paraguas = si”

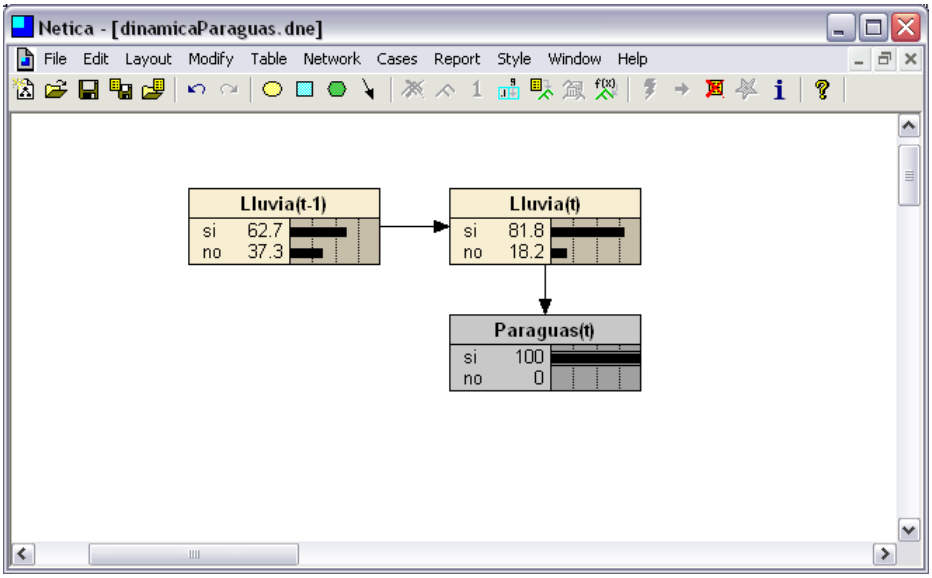

Imagen C.11. Incorporación de la evidencia “Paraguas = si”

Bibliografía

1. **Pearl, J.** *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Mateo, CA : Morgan Kaufmann Publischer, 1988.

2. **Neapolitan, R.** *Probabilistic Reasoning in Expert Systems: Theory and Algorithms.* 1990.

3. **Lauritzen, S.** *Graphilcal models.* Oxford, UK : Oxfor University Press, 1996.

4. **Jensen, F.V.** *Bayesian Networks and Decision Graphs.* Springer-Verlang, Berlín : s.n., 2001.

5. **Jordan, M. I.** *An Introducction to Graphical Models.* 2003.

6. **Patrick Murphy, Kevin.** *Dynamic Bayesian Networks: Representation, Inference and Learning.* s.l. : UNIVERSITY OF CALIFORNIA, BERKELEY, 2002.

7. **Díez, Javier.** Proyecto Elvira. *Proyecto Elvira.* [Online] Noviembre 10, 2005. [Cited: Septiembre 28, 2009.] http://www.ia.uned.es/~elvira/#proyecto.

8. *Situation Assessment via Bayesian Belief Networks.* **Das, Subrata, Grey, Rachel and Gonsalves, and Paul.** Cambridge, MA : s.n., 2002. Fifth International Conference on. pp. 664- 671.

9. *Video Understanding For Metro Surveillance.* **Cupillard, F. and Bremond F., Avanzi A. and Thonnat. M.**

10. *Bayesian Methods for Image Super-Resolution.* **Pickup, Lyndsey C., Capel, David P. and Zisserman, Stephen J.Roberts and Andrew.** 0, 2006, The Computer Journal, Vol. 0.

11. *Exploiting Human Actions and Object Context for Recognition Tasks.* **Moorey, Darnell J., Essayz, Irfan A. and III, and Monson H. Hayes.** Atlanta, Georgia : s.n., 1999. 7th IEEE International Conference on Computer Vision Corfu.

12. **Datcu, D. Rothkrantz, L.J.M.** *Automatic recognition of facial expressions using Bayesian Belief Networks.* The Netherlands : Department of Information Technology and Systems T.U.Delft.

13. *Knowledge Engineering for Large Belief Networks.* **Pradhan, M, et al.** San Francisco: Morgan Kaufmann : s.n., 1994. Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence. pp. 484-490.

14. *Bayesian user modeling for inferring the goals and needs of software users.* **Horvitz, E.J, Breese, J. S. and Heckerman, D. and Hovel,D.** Masidon, Wisconsin : Morgan Kaufman, 1998. Uncertainty in Artificial Intelligence: Proceedings of the Fourteenth Conference. pp. 256-265.

15. **Lacave Rodero, Carmen.** *Explicación en Redes Bayesianas causales. Aplicaciones Médicas.* Madrid : s.n., 2002. pp. Lacave Rodero, Carmen. 2002, Madrid.. Esvuela Superior de Informática. Universidadd de Castilla – La Mancha. [En línea] 2002, Madrid. www.inf-cr.uclm.es/www/clacave/public/tesisdefinitiva.pdf.

16. *Redes bayesianas: El regreso de la metodología probabilística formal a los sistemas expertos.* **H. Martinez, Margarita.** Octubre 1998, Revista tecnológica, pp. 45-54.

17. **A. Feigenbaum, Edwared.** *Knowledge Engineering in the 1980's.* Dept. of Computer Science, Stanford University, Stanford, CA : s.n., 1982.

18. **Giarratano, J and Riley, G.** *Sistemas Expertos: Principios y Programación.* Mexico : Thomson Editores, 2001.

19. **Samper Márquez, Juan José.** RedCientífica Ciencia, Tecnología y Pensamiento. *RedCientífica Ciencia, Tecnología y Pensamiento.* [Online] [Cited: Octubre 1, 2009.] http://www.redcientifica.com/doc/doc199908210001.html.

20. *A review of expert systems.* **Kastner, J. K and Hong, S. J.** 1984, European Journal of Operations Research, pp. 285-292.

21. **Castillo, Enrique, Gutiérrez, José Manuel and S. Hadi, Ali.** *Sistemas Expertos y Modelos de Redes Probabilísticas.* Madrid : Monografías de la Academia Española de Ingeniería, 1998.

22. **Durkin, J.** *Expert Systems: Design and Development.* Maxwell Macmillan : New York, 1994.

23. **Harmon, Paul and King, David.** *Expert System.* s.l. : John Wiley and Sons, 1985.

24. **ISO3534-1.** *Estadística. Vocabulario y símbolos. Parte 1: Términos relativos a probabilidades y estadística general.* 1993.

25. **García, Jesús.** *Razonamiento con incertidumbre. Curso 2008-2009.* Colmenarejo : s.n., 2008-2009.

26. **Diez, F. J.** *Introducción al Razonamiento Aproximado.* s.l. : Dpto. Inteligencia Artificial, UNED, 1998 . Edición revisada: mayo 2004, 1998.

27. **Bruce, G.Buchanan and H.Shortliffe, Edward.** *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Projet.* s.l. : Addison-Wesley, 1984.

28. *Fuzzy sets.* **Zadeh, L.A.** 1965, Information and Control 8, pp. 338–353.

29. **Sgafer, G.** *A Mathematical Theory of Evidence.* s.l. : Princeton University, 1976.

30. **Weiss, S. M and Kuliskowski, C. A.** *A Practical Guide to Designing Expert Systems.* s.l. : Rowman and Allanheld, Totowa, N. J, 1984.

31. **B.Korb, Kevin and E.Nicholson, Ann.** *Bayesian Artificial Intelligence.* s.l. : Chapman & Hall/CRC, 2004.

32. **Russell, Stuart y Norvig, Peter.** *Inteligencia Artificial.Un enfoque moderno.* Madrid : Pearson Prentice Hall, 2004.

33. —. *Artificial Intelligence: A Modern Approach.* Englewood Cliffs, NJ : Prentice Hall, 1995.

34. **Susi García, Rosario.** *Análisis de sensibiblidad en redes bayesianas gaussianas.* Madrid : Universidad Complutense de Madrdi, 2007.

35. **G. Cowell, Robert, et al.** *Probabilistic Networks and Expert Systems. Chapter 7 Gaussian and Mixed Discrete-Gaussian Networks.* New York : Springer, 1999.

36. **Juez Martel, Pedro and Diez Vegas, Fco. Javier.** *Probabilidad y estadistica en medicina.* s.l. : Diaz de Santos.

37. *The sensitivity of belief networks to imprecise probabilities:an experimental investigation.* **Pradham, Malcolm, et al.** 85, 1996, Artificial Intelligence, pp. 363-397.

38. **Lacave Rodero, Carmen.** *Tesis Doctoral. Explicación en redes bayesianas causales. Aplicaciones Médicas.* Madrid : Departamento de Inteligencia Artificial Universidad Nacional de Educación a Distancia, 2002.

39. **Diez, F.J and Druzdzel, M.** *Canonical probabilistic models for knowledge engineering.* Madrid : Informe técnico IA-02-01, Dpto. Intelifgencia Artificial, UNED, 2002.

40. **Gámez Martín, José Antonio and Puerta Callejón, José Miguel.** *Sistemas Expertos Probabilísticos.* Cuenca : Universidad de Castilla-La Mancha, 1998.

41. *A computational model for causal and diagnostic reasoning in reasoning in inference system.* **Kim and Pearl.** 1983. In Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI). pp. 190-193.

42. **Hernánderz Orallo, José, Ramírez Quintana, Mª Jose and Ferri Ramírez, César.** *Introducción a la Minería de Datos.* Madrid (España) : Pearson Prentice Hall, 2007.

43. **Abad, María del Mar.** *Aplicación del principio inductivo de MEVR en la construcción de clasificadores.* s.l. : Departamento de Ingeniería de la Información y las Comunicaciones, Universidad de Murcia, España, 2001.

44. *Learning limited dependence Bayesian classifiers.* **Sahami, M.** 1996. Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. pp. 335-338.

45. **Velasco Villanueva, David A.** Intelifencia Artificial II. [Online] 05 24, 2007. [Cited: 12 01, 2009.] http://www.infor.uva.es/~calonso/IAII/Aprendizaje/TrabajoAlumnos/RBmemoria.pdf.

46. **Ezequiel Felgaer, Pablo.** *Optimización de redes bayesianas basado en técnicas de aprendizaje por inducción. Tesis de grado en Ingeniería en Informática.* s.l. : Laboratorio de Sistemas Inteligentes. Facultad de ingeniería universidad de Buenos Aires, 2005.

47. *Approximating discrete probability distributions with dependence trees.* **Chow, C and Liu, C.** 1968, IEEE Trans. on Info. Theory, 14, pp. 462-467.

48. *The recovery of causal polytrees from statiscal data.* **Rebane, T and Pearl, J.** Nort-Holland : Levitt and J.F. Lemmer, 1989. In Uncertainty in artificial Intellegence, 3, L.N. Kanal, T.S.

49. *Induction of Dependency Structures from Data and its Application to Ozone Prediction.* **Sucar, L.E, Pérez-Brito, P. and Ruiz Suárez, J.C.** Melbourne, Australia : s.n., 1995. VIII International Conference on Industrial and Engineering.

50. **Sierra Araujo, Basilio.** *Aprendizaje automático : conceptos básicos y avanzados : aspectos prácticos utilizando el software Weka.* Madrid : Pearson/Prentice Hall, 2006.

51. **Fernández Galán, Severino.** *Redes bayesiana temporales: aplicaciones médicas e industriales.* Madrid : Departamento de inteligencia artificial. Facultad de ciencias. UNED., 2002.

52. **Corporation, Norsys Software.** Netica APIs. [Online] [Cited: 1 20, 2010.] http://www.norsys.com/netica_api.html.

53. **Norsys Software Corporation.** DNET-1 FILE FORMAT. [Online] [Cited: 9 22, 2009.] http://www.norsys.com/downloads/DNET_File_Format.txt.

54. *Reliability modelling with dynamic bayesian networks.* **Weber, Philippe and Lionel, Jouffe.** Whashington, D.C., USA : 5th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, 2003.

55. **Willis, Daniel.** *Ambulation Monitoring and Fall Detection System using Dynamic Belief Networks.* s.l. : School of Computer Science and Software Engineering, 2000.

56. *Software for Graphical models: A review.* **Murphy, Kevin.** December 2007, ISBA Bulletin.

57. **Corporation, Norsys Software.** Welcome to Netica's On-Screen Help. [Online] [Cited: 2 20, 2010.] http://www.norsys.com/WebHelp/NETICA.htm.

58. **Corp, Norsys Software.** *Netica Application. User`s Guide.* Vancouver, BC, Canada : Norsys Software Corp, 1996.

59. *The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering.* **A. Feigenbaum, Edward.** Cambridge : William Kaufmann, 1977. International Joint Conference on Artificial Intelligence. pp. 1014-1029.

60. **Díez, F.J.** *Sistema Experto Bayesiano para Ecocardiografía.* Dept. de Informática y Automática : Tesis Doctoral, UNED, 1994.